Emberi tevékenység felismerés EEG szenzorok adataiból gépi tanulással

Egyetemi doktori (PhD) értekezés

A szerző neve: Majoros Tamás

A témavezető neve: Dr. Oniga István

DEBRECENI EGYETEM

Természettudományi és Informatikai Doktori Tanács

Informatikai Tudományok Doktori Iskola

Debrecen, 2023.

Ezen értekezést a Debreceni Egyetem Természettudományi és Informatikai Doktori Tanács Informatikai Tudományok Doktori Iskola Informatikai rendszerek és hálózatok programja keretében készítettem a Debreceni Egyetem műszaki doktori (PhD) fokozatának elnyerése céljából.
Nyilatkozom arról, hogy a tézisekben leírt eredmények nem képezik más PhD disszertáció részét.

Debrecen, 2023……

.....
a jelölt aláírása

Tanúsítom, hogy Majoros Tamás doktorjelölt 2017-2021 között a fent megnevezett Doktori Iskola Informatikai rendszerek és hálózatok programjának keretében irányításommal végezte munkáját. Az értekezésben foglalt eredményekhez a jelölt önálló alkotó tevékenységével meghatározóan hozzájárult. Nyilatkozom továbbá arról, hogy a tézisben leírt eredmények nem képezik más PhD disszertáció részét.
Az értekezés elfogadását javaslom.

Debrecen, 2023……

.....
a témavezető aláírása

Emberi tevékenység felismerés EEG szenzorok adataiból gépi tanulással

Értekezés a doktori (Ph.D.) fokozat megszerzése érdekében az informatika tudományágban

Írta: Majoros Tamás okleveles mérnökinformatikus

Készült a Debreceni Egyetem Informatikai Tudományok doktori iskolája Informatikai rendszerek és hálózatok programjának keretében

Témavezető: Dr. Oniga István

Az értekezés bírálói:

	Dr……..	……..
	Dr……..	……..

A bírálóbizottság:

elnök:	Dr……..	……..
tagok:	Dr……..	……..
	Dr……..	……..
	Dr……..	……..
	Dr..……	……..

Az értekezés védésének időpontja: 2023. …… … .

Tartalomjegyzék

Rövidítések

ADC	Analóg-digitális átalakító (analog-digital converter)
ANN	Mesterséges neurális hálózat (artificial neural network)
BCI	Agy-számítógép interfész (brain-computer interface)
CNN	Konvolúciós neurális hálózat (convolutional neural network)
CNN1	1. számú konvolúciós neurális hálózati struktúra
CNN2	2. számú konvolúciós neurális hálózati struktúra
CNN3	3. számú konvolúciós neurális hálózati struktúra
CNN4	4. számú konvolúciós neurális hálózati struktúra
DPU	Deep learning processing unit
DSP	Digitális jelfeldolgozás (digital signal processing)
DT	Döntési fa (decision tree)
ECoG	Elektro-kortikogram
EEG	Elektroencefalográfia
EKG	Elektrokardiográfia
EMG	Elektromiográfia
FFT	Gyors Fourier-transzformáció (fast Fourier transform)
FPGA	Felhasználás helyén programozható logikai kapumátrix (field-programmable gate array)
ICA	Független komponens analízis (independent component analysis)
IP	Szellemi tulajdon (intellectual property)
kNN	k-legközelebbi szomszéd (k-nearest neighbors)
LDA	Lineáris diszkriminancia-analízis (linear discriminant analysis)
LSL	Lab Streaming Layer

LUT	Lookup table
MLP	Többrétegű perceptron (multilayer perceptron)
MSE	Átlagos négyzetes hiba (mean squared error)
NB	Naiv Bayes
PSD	Teljesítménysűrűség-spektrum (power spectral density)
RCNN	Rekurrens konvolúciós neurális hálózat (recurrent convolutional neural network)
ReLU	Rektifikált lineáris egység (rectified linear unit)
RF	Véletlen erdő (random forest)
RNN	Rekurrens neurális hálózat (recurrent neural network)
STFT	Rövid idejű Fourier-transzformáció (short-time Fourier transform)
SVM	Támogató vektor gép (support vector machine)
WT	Wavelet transzformáció

1. Bevezetés

Az agy működését kísérő elektromos jelenségek elemzésének egyik módszere elektroenkefalográfia (EEG), amelynek segítségével tanulmányozható a pszichés működés élettani háttere az idegsejtek elektromos aktivitásának regisztrálása útján. Agyi tevékenység során az agykéregben lévő neuronok aktivitásából származó ionáramok elektromos feszültégingadozásokat eredményeznek a kéreg felszínén [1]. Ez a feszültség onnan invazív vagy nem invazív úton elvezethető és megmérhető. Invazív esetben (elektro-kortikogram, ECoG) a koponyán át fúrt lyukon keresztül közvetlenül az agyszövetben helyezik el a mérőelektródákat, míg nem invazív esetben (EEG) kívül, a (hajas) fejbőrön történik az elektródák elhelyezése – emberek esetén néhány speciális kivételtől eltekintve a nem invazív eljárás használatos. Egy-egy agyi neuron működése által okozott feszültségingadozás rendkívül csekély mértékű, azonban sok neuron egy időben történő aktivitása már mérhető, néhány tíz μV nagyságrendű feszültségváltozást okoz. A mérés során kapott jelek regisztrálhatók, és így egy komplex, időben változó, az agyi tevékenységet leíró görbéhez jutunk.

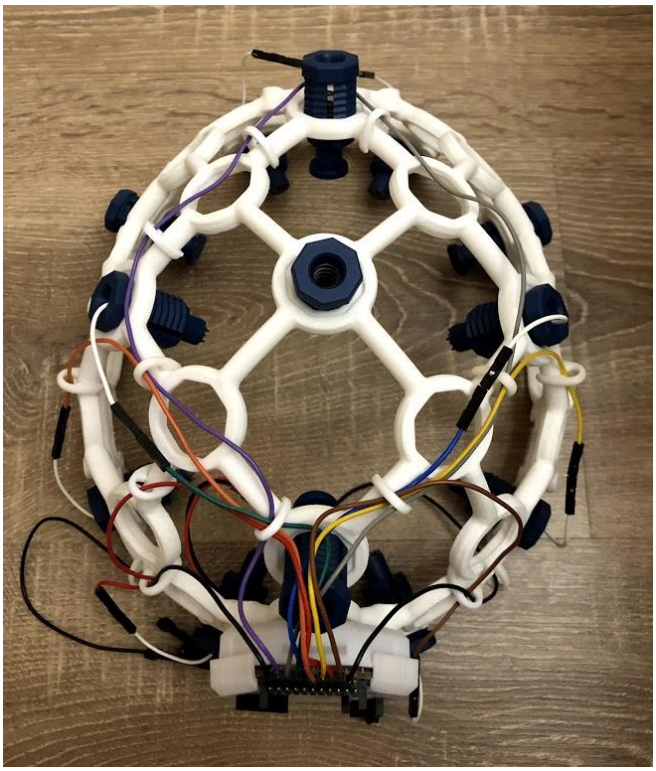
A kapott jel összetett, helyes értelmezése több évnyi tanulást és tapasztalatot igényel a szakértők részéről. Manapság azonban a gépi tanulás tudományának fejlődésével a tanulóalgoritmusok fokozatosan felváltják a bonyolult, időt és szakértelmet igénylő vizuális kiértékelést és lehetővé teszik az agyi aktivitás EEG felvételeiből információ kinyerését, így központi szerepet töltenek be számos EEG-alapú kutatásban és alkalmazásban. Például a gépi tanulási technikák sok klinikai alkalmazásra szánt EEG-alapú agy-számítógép interfész (*brain-computer interface*, BCI) központi elemét képezik mind a kommunikációban, mind a rehabilitációban [2]. A BCI célja kommunikációs kapcsolat létrehozása az emberi agy és egy számítógép között,

amelynek segítségével az agyhullámok tényleges fizikai mozgássá alakíthatók izmok használata nélkül. Ezek a rendszerek lehetővé teszik, hogy súlyosan bénult emberek kommunikáljanak [3], rajzoljanak [4], vagy akár robotokat vezéreljenek [5]. Az elmúlt években elért lenyűgöző haladás számos példája ellenére azonban továbbra is jelentős javulást lehet elérni az EEG-ből származó információk értelmezésének pontossága szempontjából. Az EEG jelek robosztus automatikus kiértékelése fontos lépés afelé, hogy ez a módszer egyre több és több alkalmazásban legyen hasznosítható és kevésbé támaszkodjon képzett szakemberekre.

Az automatikus kiértékelés (osztályozás) alkalmazása során számos megoldandó probléma, illetve kérdés merül fel. Az egyik ilyen, hogy a mérésből származó nyers adatok milyen formában, milyen esetleges előfeldolgozás után kerüljenek felhasználásra a gépi tanulási modellben. Egy másik kérdés, hogy szükséges-e jellemzők (*features*) kinyerése az adatokból, és ha igen, akkor milyen típusúak legyenek ezek. Ezek után választanunk kell a számtalan gépi tanulási módszer közül egy, a feladat megoldására alkalmasat, amely lehet akár sekély, akár mélytanuló algoritmus. A választás függhet attól, hogy hány, és milyen típusú jellemzőt nyertünk ki az adatokból, illetve milyen egyéb követelmények (pl. erőforrásigény, sebesség) merülnek fel az alkalmazandó módszerrel szemben. Végül az alkalmazott módszer paramétereit finomhangolni kell, kiértékelni a teljesítményét, és annak ismeretében további finomításokat végezni, ha szükséges, akár a jellemzők kinyerése, a választott módszer, vagy a módszer paramétereinek tekintetében.

A gépi tanulás további fejlesztésének szükségességén túl egy másik probléma az EEG rendszerek ára és kényelmetlensége. Az orvosi minőségű, laboratóriumokban használt EEG eszközök rendkívül megbízhatók és jó minőségű jelek rögzítésére alkalmasak, azonban egyrészt igen költségesek, másrészt valós, nem laboratóriumi környezetben történő használatuk

nehézkes. Az elmúlt években sorra jelentek meg a kifejezetten fogyasztói szegmensnek szánt, olcsóbb EEG eszközök, mint például a NeuroSky MindWave, az Emotiv Epoc, Emotiv Insight, vagy épp az OpenBCI Ultracortex headset. Egy ilyen, általam is használt OpenBCI Ultracortex Mark IV headset az 1. ábrán látható.



1. ábra. Méréseim során használt OpenBCI Ultracortex Mark IV EEG headset

Az előbbieken kívül piacra kerültek a közelmúltban alternatív megoldásként olyan eszközök is, melyek kizárólag a hajjal nem fedett fejbőr területeiről gyűjtenek adatokat, elkerülve az elektródák és a fejbőr között lévő haj zavaró hatásait és az ebből adódó technikai korlátokat, továbbá növelve a viselési komfortot, különösen nedves elektródák esetén [6].

A gépi tanulási módszerek alkalmazásához nagy mennyiségű adat szükséges. Ilyen adatbázisok önállóan is készíthetők, de ez nehézkes, hiszen

fejlett EEG szenzorokat, adatgyűjtő rendszert és számos önkéntest igényel. Az EEG-hez kapcsolódó kutatások töretlen népszerűségének köszönhetően azonban megjelentek olyan nyilvános adatbázisok is, amelyek lehetővé teszik nagyszámú páciens adatainak elemzését. Ezek az adatbázisok különböző célokból készülnek, például epilepszia [7], alvási rendellenességek [8-9] vagy épp motoros tevékenységek végzése során az agyban lejátszódó folyamatok [10] vizsgálhatók segítségükkel.

Az általam végzett kutatás célja ezen tudományos terület további fejlődésének elősegítése volt, mind a gépi tanulás terén, mind pedig megfizethető árú EEG adatgyűjtők tevékenységfelismerés céljából történő felhasználási lehetőségének vizsgálatával. Ehhez részben nyilvánosan elérhető EEG adatbázisokat, részben pedig saját EEG eszközzel mért adatokat használtam fel. További célként tűztem ki a hardveres implementációs lehetőségek vizsgálatát, illetve egy saját, automatizált megoldás elkészítését, amellyel egy ilyen hardveres neurális hálózat létrehozható.

2. Irodalmi áttekintés és előzmények

Az EEG jelekből gépi tanulás útján történő tevékenységfelismerés számos kihívást tartogat. Ezek közül az egyik az EEG rossz jel-zaj viszonya, ugyanis a tényleges, hasznos jelre a legtöbb esetben szuperponálódik valamilyen, akár külső forrásból származó zaj. Ilyenek például a pislogásból, különböző mozgásokból, izomtevékenységből vagy az elektromos hálózatból származó zajok (*artifact*), amelyeket a feldolgozás megkönnyítése érdekében érdemes eltávolítani. A [11-12] cikk szerzői különböző automatizált módszereket dolgoztak ki a jel-zaj viszony javítása, illetve a túl zajos vagy hibás csatornák kiszűrése céljából.

Egy másik felmerülő probléma az egyének közötti fiziológiás különbségek, amelyek jelentősen befolyásolni tudják egy gépi tanulási modell teljesítményét: egy modell, amely jól teljesít egy bizonyos egyén EEG jeleinek felismerésében, könnyen adhat nagyon rossz eredményt egy másik alany esetében. Mivel kulcsfontosságú az, hogy egyének egy halmazának adataiból egy másik, ismeretlen halmazra is tudjunk általánosítani, ezért jelentős erőfeszítéseket tesznek olyan módszerek kifejlesztésére, amelyek képesek kezelni ezeket az egyének közti különbségeket. A legmodernebb technikák, mint például a Riemann-féle geometrián alapuló osztályozók [13] és az adaptív osztályozók [14] változó sikerrel képesek kiküszöbölni ezeket a problémákat.

Az EEG jelekből történő tevékenységfelismerés alapötlete az, hogy tevékenységek végzése közben az agy generál olyan mintákat, amelyek csak az adott tevékenységre jellemzők, így a különböző tevékenyégek ezen minták alapján egymástól elkülöníthetők az EEG-ben. Erre a célra számos gépi tanulási módszer alkalmazható, ideértve a sekély és mélytanulási technikákat is. Az egyik ilyen sekély gépi tanulási módszer az SVM (*support vector machine*) alkalmazása, amely lineárisan elválasztható csoportok

osztályozására szolgál olyan módon, hogy meghatározza azt a szeparáló hipersíkot, amely esetén a legnagyobb a margó.

A k-legközelebbi szomszéd (*k-nearest neighbors,* kNN) módszerrel történő osztályozás esetén a tesztvektor valamilyen metrika alapján (pl. euklideszi távolság) meghatározott *k* legközelebbi szomszédját vesszük a tanító halmazból, ezek közül a hozzájuk kapcsolódó osztálycímkék között legtöbbször előfordulót rendeljük a tesztadathoz.

Döntési fa (*decision tree,* DT) esetén a nemterminális csúcsok tesztfeltételt tartalmaznak. A gyökércsúcsból indulva vizsgáljuk, hogy a tesztesetre igazak-e az egyes feltételek, ilyen módon bejárva a fát, amíg végül egy levélcsúcshoz érünk. A levélcsúcsokban már nem feltételt, hanem egy osztálycímkét találunk, így megkapjuk, hogy melyik osztályba tartozik a vizsgált minta.

A véletlen erdő (*random forest,* RF) módszer a döntési fák kiterjesztése olyan módon, hogy több különböző, egymástól független döntési fát hoz létre a tanulás során, amelyek mindegyike hoz egy döntést és ezek közül a legtöbbször előforduló osztályt rendeljük a tesztesethez. Az alapötlet ebben az esetben, és az ehhez hasonló együttes tanulási módszereknél az, hogy gyenge osztályozók csoportba szervezve, kollektív módon egy erős, hatékony tanulóalgoritmussá válhatnak.

A naiv Bayes (NB) osztályozó megbecsüli az osztályra vonatkozó feltételes valószínűségeket, azt feltételezve, hogy adott osztály mellett az attribútumok feltételesen függetlenek egymástól, majd osztályozáskor az így kapott feltételes valószínűségek felhasználásával adja meg a legvalószínűbb osztályt.

A neurális hálózatoknak (*artificial neural network*, ANN) számos típusa létezik, közülük az egyik legegyszerűbb, de gyakran alkalmazott hálózat a

többrétegű perceptron (*multilayer perceptron,* MLP), amely egy előrecsatolt neurális hálózat. Ez legalább három rétegből áll (bemeneti, kimeneti és egy vagy több rejtett réteg), a rétegekben neuronok találhatók, a bemeneti rétegben lévő neuronok kivételével aktivációs függvénnyel együtt. Az egymást követő rétegek teljesen összekapcsoltak, azaz bármely rétegben lévő összes neuron kapcsolódik az azt követő réteg összes neuronjához.

A mélytanulási módszerek megjelenését követően azok egyre elterjedtebbé váltak a legkülönfélébb gépi tanulási problémák esetében. Az említett MLP hálózat több rejtett réteg alkalmazásával már a mélytanulási módszerekhez sorolható, azonban bizonyos kiegészítésekkel ettől komplexebb mélytanuló hálózatok is létrehozhatók. A rekurrens neurális hálózat (*recurrent neural network*, RNN) például az MLP-vel ellentétben nem csak előrecsatolást, hanem visszacsatolást is tartalmaz, amely tulajdonképpen memóriával látja el a hálózatot. Konvolúciós neurális hálózat (*convolutional neural network,* CNN) esetén pedig új típusú rétegekkel egészül ki a hagyományos, csak neuronokat tartalmazó előrecsatolt hálózat. Ezek az új rétegek képesek automatizálni a sekély módszerek esetén „kézzel" történő jellemző kinyerést, ilyen módon egy általánosabban használható megoldást biztosítanak. Az előbbi két megoldásnak, azaz a visszacsatolásnak és a konvolúciós rétegek hozzáadásának a kombinációja is lehetséges, ekkor rekurrens konvolúciós neurális hálózatról (RCNN) beszélünk.

Számos kutató vizsgálta az előbb említett (és egyéb) gépi tanulási módszerek alkalmazhatóságát EEG alapján történő tevékenységfelismerésben, azonban a kapott eredmények nem támasztják alá, hogy lenne egy olyan algoritmus, amelynek hatékonysága egyértelműen jobb a többinél. Például a [15] cikk szerzői öt sekély algoritmust használtak elképzelt motoros tevékenység felismerésére kilenc önkéntes esetén. Négy alanynál a naiv Bayes, kettőnél a DT, kettőnél a kNN, egynél az SVM bizonyult a leghatékonyabbnak.

A [16] cikk szerzői egy, az előzőhöz hasonló adatbázison SVM és CNN osztályozók közül mind a kilenc alany esetén a CNN-t találták pontosabbnak. Ezzel szemben viszont a [17] tanulmányban kilencből öt alany esetén az SVM adott jobb eredményt a CNN-nel szemben.

A [18] cikk szerzői elképzelt motoros tevékenységek felismerésére alkalmaztak MLP, CNN és RNN típusú hálózatokat. Eredményeik alapján a CNN bizonyult a három közül a legjobbnak, továbbá megmutatták, hogy egy ugyanolyan típusú, de nagyobb, több rétegű modell nem feltétlenül jobb, mint egy sekélyebb, azaz a hálózat komplexitása nem korrelál a felismerési pontosságával. Ezen túl rámutattak arra is, hogy a CNN hálózatok teljesítményét nagymértékben befolyásolja a hiperparaméterek, azaz például a kernelméret és kernelszám megválasztása. A [19] publikációban a szerzők szintén RNN és CNN algoritmust vizsgáltak, és megállapították, hogy ebben az alkalmazásban a CNN jelentősen felülmúlja az RNN-t.

A [20] tanulmányban a kutatók egy öt önkéntestől származó EEG adatbázist használtak, amelyen a jobb kéz és jobb lábfej elképzelt mozgatását próbálták elkülöníteni egymástól. Ehhez DT, MLP, SVM, kNN, NB és RF algoritmusokat használtak zajcsökkentést, jellemző kinyerést és dimenziócsökkentést követően. Az elért osztályozási pontosság szempontjából az NB 53%-os eredménye bizonyult a legrosszabbnak, amelytől jelentősen jobban teljesítettek a DT (64%), MLP (67%), RF (78%) és SVM (89%) módszerek, de a legjobb eredményt a kNN algoritmus nyújtotta közel 95%-os pontosságával az öt önkéntes átlagában. Megjegyzendő ugyanakkor, hogy volt olyan alany, akinek adatai esetén a DT és az RF algoritmus felülmúlta ezt az eredményt 95%, illetve 98%-os osztályozási pontosságával.

A [21] cikk szerzői szintén SVM és MLP algoritmust használtak elképzelt motoros tevékenységek felismerésére, azonban a [20] cikkel

ellentétben ők hatékonyabbnak találták az MLP-t: SVM esetén 75%, MLP esetén 80% osztályozási pontosságot sikerült elérniük.

Az előbbiekben hivatkozott kutatások rámutattak arra, hogy korántsem egyértelmű, hogy milyen gépi tanulási módszer alkalmazása lehet a leghatékonyabb az EEG alapján történő tevékenységfelismerésben. Bizonyos esetekben sekély, más esetekben mélytanuló algoritmusok bizonyultak pontosabbnak az osztályozásban. Ha egyébként ezek a kutatások nem mutattak volna olykor egymásnak ellentmondó eredményeket, akkor sem lehetne elvégezni ezek alapján egy sorrend felállítását az egyes algoritmusok között, hiszen azok eltérő architektúrájúak voltak, eltérő módon előfeldolgozott adatokon, eltérő adatbázisokon lettek alkalmazva, így általános érvényű konklúzió levonása nem lenne lehetséges. Mindezek mellett ugyanakkor megfigyelhető az a tendencia, hogy ebben a kutatási témában az utóbbi években a legáltalánosabban elterjedt algoritmussá a konvolúciós neurális hálózat vált [22].

Az EEG jelek bonyolultak és nagy mennyiségű információt tartalmaznak. Az említett tanulmányok alapján látszik, hogy egy hálózat hatékonyságában nagy szerepe van a megfelelő algoritmus és architektúra kiválasztásának, azonban legalább ennyire befolyásolni tudja a végeredményt az adatok előfeldolgozása és a jellemzők kinyerése. A jellemzők kinyerésének célja az adatok alacsonyabb dimenziós térbe transzformálása úgy, hogy az megtartja az EEG jelek által közvetített kritikus információkat [23]. A szakirodalomban számos jellemzőkinyerési módszert javasoltak a konkrét feladat alapján, ideértve az időtartományt, frekvenciatartományt, idő-frekvencia tartományt, valamint a térbeli kapcsolatokat [24].

A [22] kutatás széleskörű áttekintést ad a különböző EEG-n történő mélytanulást vizsgáló cikkekről. Ez alapján CNN használatánál a számba vett

cikkek több, mint 55%-ánál a rögzített jelek értéke közvetlen módon, szűk 30%-ánál képpé konvertálva szolgált a hálózat bemeneteként, és csupán az esetek körülbelül 15%-ában használtak fel valamilyen, adatokból kinyert jellemzőt. Érdemes azt is megemlíteni, hogy ez utóbbi esetekben a kutatók által elért pontosság átlaga 84%, míg a jelek közvetlen használatánál 87% volt, amely cáfolja azt a feltételezést, hogy minél több erőfeszítést teszünk az adatok jobb előfeldolgozásáért, annál pontosabb lesz az osztályozás. Sőt, egyenesen arra a meglepő következtetésre mutat rá, hogy a neurális hálózatra bízva ezt a feladatot jobb végső eredményt érhetünk el. Ezek a megfigyelések összhangban vannak azzal a ténnyel, hogy a konvolúciós rétegek képesek az automatikus jellemzőkinyerésre, és megmutatták azt, hogy emellett további statikus módszer alkalmazása nem indokolt. Szintén ezen cikk alapján viszont MLP használatakor egy kivétellel (ahol közvetlenül a rögzített jelek értékét használták) minden esetben kinyert jellemzők felhasználásával történt az osztályozás.

Az időtartományból kinyert jellemzők közé soroljuk a különböző statisztikai mérőszámokat, mint például az átlag, a szórás, az átlagos abszolút eltérés, a négyzetes közép, a kurtózis, vagy épp a minimum-maximum közti különbség. A frekvenciatartománybeli jellemzők között megemlíthetjük például a spektrum energiát (frekvenciakomponensek abszolút értékének négyzetösszege), a spektrum entrópiát, a spektrumközepet (a spektrum várható értéke, ha azt valószínűségi eloszlásként tekintjük), vagy az alapfrekvenciát, amely a jelben megtalálható legnagyobb amplitúdójú komponens frekvenciája.

Bár a témában egyre több cikk jelenik meg, továbbra is nyitott kérdés, hogy milyen jellemzők felhasználásával maximalizálható a gépi tanulás pontossága. A [25] tanulmányban a kutatók mintegy egy tucat különböző idő- és frekvenciatartománybeli jellemzőt nyertek ki az EEG adatokból, amelyeket lineáris diszkriminancia-analízis (*linear discriminant analysis,* LDA)

módszerrel történő osztályozáshoz használtak fel. Közülük a szerzők egy frekvenciatartománybeli jellemzőt, a teljesítménysűrűség-spektrumot (p*ower spectral density,* PSD) találták a leghatékonyabbnak.

Az előző kutatás nem vett számba idő-frekvenciatartománybeli jellemzőket, annak ellenére, hogy a frekvenciatérbeli leírás önmagában semmit nem mond a jel időbeli változásairól, így kevesebb információt tartalmaz, mint az idő-frekvenciatérbeli leírás. Rövid idejű Fourier-transzformációval (*short-time Fourier transform,* STFT) vagy wavelet transzformációval (WT) a frekvenciaspektrumban történő változások azok előfordulási idejével együtt határozhatók meg, amelyet kinyert jellemzőként használva felveti az osztályozó algoritmus pontosságának további növelési lehetőségét. A [26] cikk szerzői öt különböző matematikai módszert alkalmaztak frekvencia- és idő-frekvenciatartománybeli jellemzőkinyerésre. Ezek a gyors Fourier-transzformáció (*fast Fourier transform*, FFT), wavelet transzformáció, sajátvektorok, idő-frekvencia eloszlás és autoregresszív eljárások voltak. Eredményeik azt mutatták, hogy minden módszernek vannak előnyei és hátrányai a többihez képest, így az idő-fekvenciatérbeli jellemzők használata nem feltétlenül előnyösebb a frekvenciatérbeli jellemzők használatánál.

A gépi tanulási módszerek EEG jelek feldolgozásában való egyre nagyobb mértékű elterjedésével párhuzamosan egymás után jelentek meg a nyilvánosan elérhető, EEG méréseket tartalmazó adatbázisok. Ezek közül néhány, valódi vagy elképzelt motoros tevékenységek végzése során rögzített adatokat összegyűjtő adatkészlet került felsorolásra az 1. táblázatban.

1. táblázat. Motoros tevékenységek EEG felvételeit tartalmazó nyilvános adatbázisok

Referencia	**Adatgyűjtésben résztvevő önkéntesek száma**	**Tevékenységek típusa**
[27-28]	109	Kéz, lábfej mozgatása, valódi és elképzelt
[29]	12	Fogás és emelés, valódi
[30]	13	Kéz, láb, nyelv, ujjak mozgatása, elképzelt
[31]	7	Kéz, lábfej mozgatása, elképzelt
[32] 2a	9	Kéz, láb, nyelv, ujjak mozgatása, elképzelt
[32] 2b	9	Kezek mozgatása, elképzelt
[33]	14	Kéz, lábfej mozgatása

Részben egy már meglévő adatbázis használatának magától értetődő előnyei miatt, részben annak érdekében, hogy saját kutatási eredményeim összehasonlíthatóak legyenek más, korábbi publikációkban közölt eredményekkel, munkám során én is nagyrészt egy ilyen adatbázison, konkrétan a táblázat első sorában hivatkozott, 109 önkéntes részvételével készült PhysioNet adatbázison dolgoztam. Emellett bizonyos esetekben saját EEG eszközzel, saját mérőrendszerrel rögzített adatokat használtam, a mérést és kiértékelést végző programokat kifejezetten ehhez a kutatáshoz fejlesztettem.

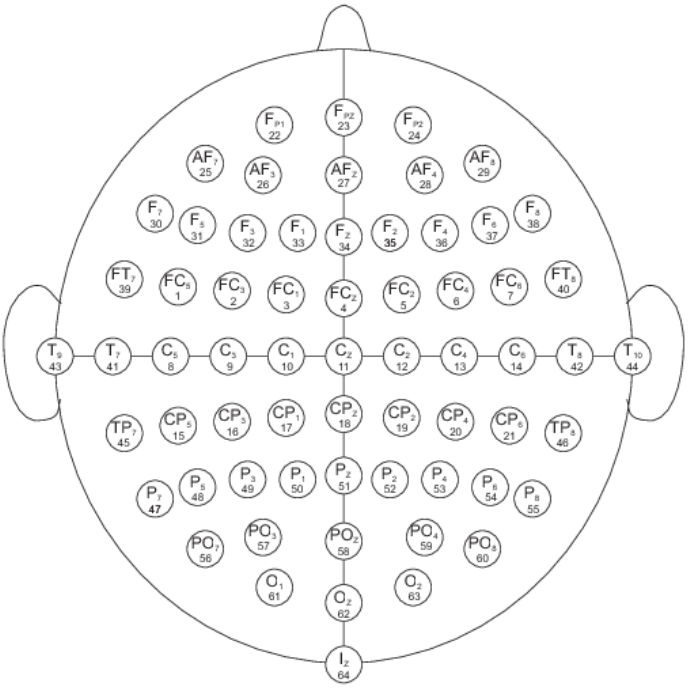
A PhysioNet adatbázis több, mint 1500 egy- és kétperces EEG felvételt tartalmaz 109 önkéntestől. A méréseket a BCI2000 rendszerrel végezték 64 csatornán, miközben az önkéntesek különböző motoros és elképzelt motoros tevékenységeket hajtottak végre. Minden alany esetén 14 mérés készült,

amelyek közül két egyperces felvétel során az önkéntesek nem végeztek tényleges tevékenységet, csupán nyitott, illetve zárt szemmel ülve rögzítették EEG jeleiket, a további 12 mérés során pedig a következő feladatokat hajtották végre egy képernyő előtt ülve, mindegyiket három alkalommal megismételve, két-két perc erejéig:

- Egy objektum megjelenik a képernyő bal vagy jobb oldalán. Az önkéntes összecsukja a megfelelő oldali öklét, amíg az objektum el nem tűnik, majd ellazul.
- Egy objektum megjelenik a képernyő bal vagy jobb oldalán. Az önkéntes elképzeli, hogy összecsukja a megfelelő oldali öklét, amíg az objektum el nem tűnik, majd ellazul.
- Egy objektum megjelenik a képernyő tetején vagy alján. Az önkéntes összecsukja mindkét öklét (ha felül jelenik meg), vagy mindkét lábfejét (ha alul jelenik meg), amíg az objektum el nem tűnik, majd ellazul.
- Egy objektum megjelenik a képernyő tetején vagy alján. Az önkéntes összecsukja mindkét öklét (ha felül jelenik meg), illetve mindkét lábfejét (ha alul jelenik meg), amíg az objektum el nem tűnik, majd ellazul.

Az objektumok megjelenítése a képernyőn négy másodpercig történt, a megjelenítések közti szünet szintén négy másodperc időtartamú volt. Az adatok 160 Hz mintavételi frekvenciával lettek rögzítve, EDF+ formátumban, amely egy széles körben elfogadott szabvány EEG adatok tárolására. Előnye az EDF-fel szemben, hogy támogatja a szabványos elektródanevek használatát és az időbélyegzett annotációkat az események tárolására [34], amelyek jelen esetben a tevékenységváltást jelentik.

Az adatbázisban az elektródák elnevezése a nemzetközi 10-10 rendszer szerint történt, kihagyva az Nz, F9, F10, FT9, FT10, A1, A2, TP9, TP10, P9 és P10-et. A használt elektródák neve és elhelyezkedése a 2. ábrán látható.



2. ábra. A PhysioNet adatbázis használt EEG elektródák [27-28]

A 2. táblázat bemutat néhány, más kutatók által a PhysioNet adatbázison elért eredményt. Megjegyzendő ugyanakkor, hogy azonos adatbázis használata esetén sem minden esetben hasonlíthatók össze az elért osztályozási pontosságok, ugyanis egyrészt számít, hogy hány önkéntes adatai lettek ténylegesen felhasználva a 109-ből, másrészt, hogy hány osztályt különböztetünk meg. Bármely mérési fájl esetén – kivéve a nyitott-csukott szem esetét – alapvetően háromféle tevékenységről állnak rendelkezésre adatok (például az első esetben bal kéz mozgása, jobb kéz mozgása, pihenés), azonban egyrészt ez szűkíthető is két osztályra, ha például a pihenést nem

kívánjuk figyelembe venni, csak az aktív tevékenységeket, másrészt bővíthető is több különböző típusú mérési fájl összevonásával.

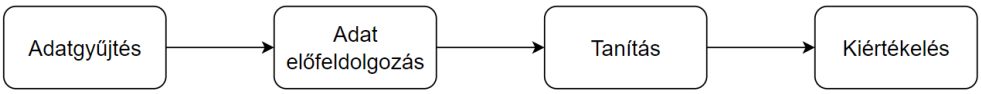
2. táblázat. PhysioNet adatbázison elért korábbi eredmények

Referencia	**Legjobb elért felismerési arány**
[35] (2 osztály)	80,05%
[36] (2 osztály)	80,1%
[36] (3 osztály)	69,72%
[36] (4 osztály)	59,71%
[37] (2 osztály)	74,71%
[38] (5 osztály)	70,64%
[39] (2 osztály)	74,9%
[40] (2 osztály)	83,26%

Kutatásomban egyrészt arra kerestem a választ, hogy el tudok-e érni a korábbiaknál jobb eredményt egy megfelelő gépi tanulási algoritmus és jellemzőkinyerési módszer megválasztásával a PhysioNet adatbázison, másrészt, hogy saját mérési adataimon mennyire hatékonyan tudok osztályozást végezni. Ehhez igyekeztem felhasználni azokat a jellemzőkinyerési módszereket, amelyekkel korábbi kutatásokban biztató eredményeket sikerült elérni vagy a feladathoz valamilyen szempontból optimálisnak gondoltam, és ezeket felhasználni olyan gépi tanulási módszerekhez, amelyek – szintén korábbi kutatások alapján – jó választásnak bizonyulhatnak EEG jelek osztályozásához. Harmadrészt pedig megvizsgáltam a neurális hálózatok felismerési sebességének hardveres gyorsítási lehetőségeit FPGA (*field-programmable gate array*) felhasználásával.

3. Adatgyűjtés

A gépi tanulás teljes folyamata egy láncnak tekinthető, melynek fő részei az adatok begyűjtése, előfeldolgozása, a kiválasztott gépi tanulási modell tanítása, végül pedig a teljesítményének kiértékelése. A folyamat fő lépései a 3. ábrán láthatóak.



3. ábra. A gépi tanulás folyamata

Ezek alapján az első lépést az adatok összegyűjtése képezi. Kutatásom nagy részében az említett nyilvános PhysioNet adatbázist használtam, hiszen így lényegesen több és változatosabb adattal dolgozhattam, mintha saját adatkészletet készítettem volna, de emellett saját méréseket is végeztem az általam összeállított mérőrendszerrel.

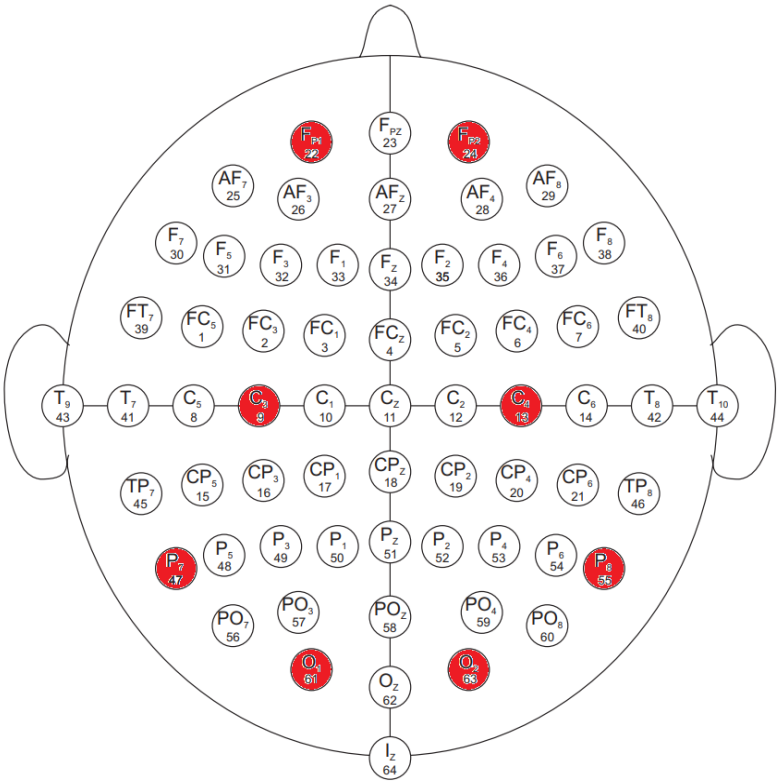
Az EEG adatok gyűjtéséhez az OpenBCI-tól származó Ultracortex Mark IV típusú EEG sisakot használtam. Az OpenBCI kutatókból, mérnökökből, művészekből és tudósokból álló közösség, bioérzékelő rendszereik szívműködés (elektrokardiográfia, EKG), agyi elektromos (EEG) és izomtevékenység (elektromiográfia, EMG) mérésére használhatók. Az Ultracortex Mark IV száraz EEG érzékelők segítségével 8 vagy 16 csatornán képes rögzíteni az agyi aktivitást, akár 35 különböző helyről. Számomra a nyolccsatornás változat állt rendelkezésre. Az eszköz jó teljesítményt és nagy pontosságot biztosít az árához mérten. A [41] cikkben kilenc résztvevő segítségével vizsgálták a sisak használhatóságát hat másik eszköz mellett, ahol a második legjobb osztályozási pontosságot nyújtotta, azonban a nála pontosabb eszköz lényegesen drágább volt. A publikáció szerzői a bekerülési költség és az elért pontosság hányadosából számított költséghatékonysági

arány alapján az OpenBCI eszközét találták a legjobbnak. A [42] cikk szerzői az eszköz működését vizsgálták a gamma hullámsávban egy professzionális termékkel összevetve. Ez a tartomány azért fontos, mert a frekvencia növekedésével a komponensek abszolút teljesítménye a jelben az „1/f" fordított arányosság alapján csökken, így a legmagasabb frekvenciájú EEG sáv lévén itt a legkisebbek a hasznos jelek, miközben jelentős zaj származik az 50 Hz-es elektromos hálózatból. A kutatás során a jel ugyan észrevehetően zajosabb volt, mint a professzionális eszköz esetén, azonban ezzel együtt is használható alternatívának ítélték meg a publikáció szerzői.

A sisak a Cyton kártyával érkezett, de bármely más OpenBCI kártyával működik. Ezen az eszközön nyolc bemenet található, amelyek bármelyike használható EEG, EMG vagy EKG mintavételre. A szenzorokból származó jelek digitalizálását egy alacsony zajszintű Texas Instruments ADS1299 analóg-digitális átalakító (ADC) végzi, amelynek felbontása 24 bit, erősítése pedig programozható. Az erősítés lehetséges értékei: 1, 2, 4, 6, 8, 12, 24. A mintavételezési frekvencia 250 Hz mind a nyolc csatornán. Az ADC mellett egy 32 bites processzor végzi a digitalizált értékek továbbítását a számítógép felé. A kényelmesebb használhatóság érdekében az eszköz lítium-ion akkumulátorról üzemeltethető és az adatgyűjtő számítógéppel történő kommunikáció vezeték nélküli kapcsolaton történik.

Az agy különböző területei különböző funkcionalitással bírnak. A prefrontális kéreg a végrehajtó funkciókban vesz részt, a parietális lebeny a szenzoros információkat integrálja, a nyakszirti lebeny pedig a vizuális információkat dolgozza fel. Az EEG elektródák fejen történő elhelyezésének (egyik) nemzetközileg elfogadott szabványa a 10-20 rendszer, az Ultracortex sisak esetén is ezen a rendszeren alapszik az érzékelők elhelyezkedése. A nyolc szenzort úgy szereltem be, hogy közülük kettő a prefrontális kéreg területén (Fp1, Fp2), kettő az agykéreg középső területein (C3, C4), kettő a parietális

lebenynél (P7, P8), kettő pedig a nyakszirti lebeny területén (O1, O2) végezzen méréseket. Az általam használt csatornák helyzete, számozása és elnevezése a 4. ábrán pirossal jelölve és a 3. táblázatban került összegzésre. Annak érdekében, hogy a PhysioNet adatbázison betanított tanulóalgoritmus a későbbiekben a saját adataim osztályozására is használható legyen, ezért abból az adatbázisból is csak az ezen a nyolc csatornán történt méréseket vettem figyelembe.



4. ábra. Saját mérések során használt csatornák elhelyezkedése

3. táblázat. Az általam használt csatornák száma és elnevezése

Csatorna száma	Csatorna neve
1. csatorna	Fp1
2. csatorna	Fp2
3. csatorna	C3
4. csatorna	C4
5. csatorna	P7
6. csatorna	P8
7. csatorna	O1
8. csatorna	O2

Az adatgyűjtéshez és a mérés irányításához kifejlesztettem egy programot, amellyel lehetőség van ahhoz hasonló jellegű mérések lebonyolításra, mint amilyeneket a PhysioNet adatbázis készítői végeztek. Az eszközről érkező adatok LSL (*Lab Streaming Layer*) protokollon keresztül továbbítódnak az adatgyűjtő szoftverhez, amely egy széles körben használt rendszer a kutatási kísérletekben. Az LSL helyi hálózaton fut, és hatékonyan kapcsolja össze a fiziológiai, illetve viselkedési adatszolgáltató rendszereket az adatmegjelenítést, rögzítést vagy elemzést végző adatfogyasztókkal [43].

Az adatgyűjtő szoftvert Matlab-ban készítettem el. A program képes az OpenBCI eszközről érkező adatfolyam fogadására, rögzítésére és felcímkézésére. Az alkalmazás véletlenszerűen megjelenít egy objektumot a képernyő felső vagy alsó részén adott időtartamig, majd az objektum eltűnik bizonyos időre és ez a folyamat ismétlődik, ugyanúgy, mint a PhysioNet mérések során. Eközben a beérkező mérési adatok rögzítésre és felcímkézésre kerülnek. A címke megadja, hogy milyen feladatot kellett végeznie az önkéntesnek az adott pillanatban.

Az alkalmazásban felhasználóbarát módon, grafikus felületen lehet a mérési paramétereket beállítani. Ezek a mérési paraméterek megadják, hogy hány másodpercig kell az objektumot megjeleníteni, hány másodpercig nem, és ez a ciklus hányszor ismétlődjön meg. Alapbeállításként a PhysioNet adatbázisban használt értékeket használtam (négy másodperc tényleges tevékenység, négy másodperc pihenés, 15-ször ismételve). Az általam készített szoftver felhasználói felülete az 5. ábrán látható.

5. ábra. Az általam készített szoftver felhasználói felülete

1. tézis: *Kifejlesztettem egy szoftvert, amely képes tetszőleges OpenBCI EEG eszközökről adatgyűjtést végezni. A szoftver egy objektumot jelenít meg véletlenszerűen a képernyő különböző részein, miközben LSL protokollon keresztül fogadja, rögzíti és felcímkézi az EEG eszközről érkező adatfolyamot. A működés helyességének igazolása laboratóriumi vizsgálatokkal történt.*

4. Adat előfeldolgozás

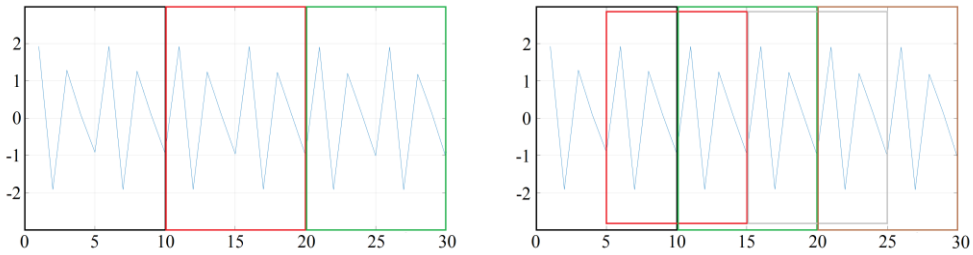
Az előző fejezetben említett gépi tanulási láncban az adatgyűjtést követő lépés azok előfeldolgozása. Ez az előfeldolgozási folyamat jellemzően további részfolyamatokra bontható, amelyek magukba foglalhatják az adatfolyam szegmentálását, jellemzők kinyerését, az adatok szűrését, feljavítását, vagy valamilyen jellegű átalakítását.

4.1. Szegmentálás

EEG jelek alapján történő tevékenységfelismeréskor a mért adatok tipikusan egy hosszú, digitalizált adatfolyamként állnak rendelkezésre, amelynek során az alany akár több különböző tevékenységet is végezhet. A tanítás során arra a feltételezésre építünk, hogy az adatokban megtalálható valamilyen mintázat, amely csak egy adott tevékenység esetén jelenik meg, ennek megfelelően legalább a tevékenységváltások bekövetkezésekor szükség van az adatfolyam széttördelésére. Jellemzően azonban a jobb teljesítmény érdekében ettől kisebb darabokra történő tördelés optimális. Az adatfolyam ilyen jellegű feldarabolását szegmentálásnak, az egyes szegmenseket pedig ablakoknak nevezzük.

A szegmentálás történhet az adatfolyam egyszerű széttördelésével (ekkor például egy tíz másodperces mérést tíz, egy másodperces darabra bontunk), de megvalósulhat csúszó ablak alkalmazásával is, ebben az esetben van valamekkora átfedés az egymást követő ablakok adatai között. Természetesen ez utóbbi esetben – az ablakok közötti átfedés mértékének függvényében – nagyobb számú tanítási és tesztelési mintához jutunk. A két módszer közti különbséget a 6. ábra szemlélteti. Mindkét módszernél ugyanazt a 30 elemből álló adatsort szegmentáltam 10-es ablakméret használatával, de az ábra bal oldalán az ablakok egymást követték, így végül 3 darab mintához

jutottunk, míg a jobb oldalon csúszó ablakozást alkalmaztam 50%-os átfedéssel, amellyel ilyen módon 5 darab mintát kaptunk. A minták ez utóbbi esetben fekete, piros, zöld, szürke, barna sorrendben követik egymást.



6. ábra. Szegmentálási módszerek

Minél nagyobb az átfedés, annál több mintát tudunk a tanításnál felhasználni, azonban túlzottan nagy mértékű átfedés esetén az egymást követő ablakok csak minimális plusz információt adnak egymáshoz képest, aminek következtében nem igazán járulnak hozzá a gépi tanulóalgoritmus pontosságának javításához, viszont a jóval nagyobb adathalmaz következtében a tanítás ideje megnő.

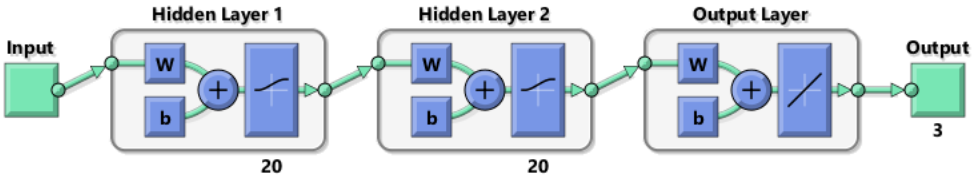
Az ablakok mérete jellemzően néhány másodperces intervallumot fed le [44-46]. Fontos az ablakméret megfelelő megválasztása, ugyanis létezik egy olyan optimális érték, amellyel az adott gépi tanulási feladat esetén a modell teljesítménye maximalizálható. Ettől kisebb, azaz túl rövid időintervallumot magában foglaló ablakméret mellett előfordulhat, hogy az ablak nem tartalmaz elég információt a végzett tevékenységről, amely így az osztályozás pontosságát csökkenti. Nagy ablakméret esetén pedig több különböző tevékenységből származó adatot is tartalmazhat, különösen, ha a tevékenységváltások viszonylag gyakoriak. Bár ez utóbbi probléma egyszerűbben orvosolható, hiszen eldobhatjuk azokat az ablakokat, amelyek közben tevékenységváltás történt, de ha túl sok ilyen van, az a rendelkezésre álló tanítási minták számának szignifikáns csökkenését okozhatja. Egy másik

probléma pedig már nem a tanítás során, hanem a betanított modell használatakor jelentkezik nagyméretű ablak esetén: valós idejű tevékenységfelismeréskor a végzett tevékenység megváltozását követően nagyobb késleltetéssel jelenik meg a kimeneten az osztályozás eredménye, ráadásul ez alatt a hosszabb késleltetés alatt a kimenet egyébként sem megbízható, hiszen az ablakban több különböző tevékenységből származó adat van.

Kutatásomban a PhysioNet adatbázis felhasználásával igyekeztem megtalálni az ideális ablakméretet. Ehhez az adatbázis egyes sorszámú önkéntesének egy elképzelt motoros tevékenység végzése közben rögzített adatfájlját használtam: az önkéntes előtti kijelzőn véletlenszerűen vagy felül vagy alul megjelent egy objektum és ennek megfelelően vagy mindkét öklének, vagy mindkét lábfejének összecsukását képzelte el, az objektum eltűnésekor pedig pihent. A váltások négy másodpercenként történtek. Az adatok szegmentálását különböző ablakméretekkel végeztem. Az ablakok majdnem teljesen átfedőek voltak, egy *N* méretű ablak az aktuális és az előző *N-1* mérési pontot tartalmazta a korábban említett nyolc darab EEG csatorna adataiból. Az ablakok sorrendjének véletlenszerű átrendezését követően csatornánként elvégeztem a teljesítménysűrűség-spektrum meghatározását, majd a spektrum integrálását külön-külön az alfa, béta, gamma, delta és théta sávokban. Ebből következőn ablakonként 40 érték állt rendelkezésre (8 csatorna, csatornánként 5 sáv), ezeket használtam egy MLP neurális hálózat tanítására. A hálózat felépítését tekintve két rejtett réteget tartalmazott, rétegenként 20-20 neuronnal, a neuronok aktivációs függvénye szigmoid típusú volt.

Ezen paraméterek, továbbá a teljesítménysűrűség-spektrum kinyert jellemzőként való használatának okáról, illetve az EEG sávokról a dolgozat

későbbi részében esik szó. Az alkalmazott neurális hálózat blokkvázlata a 7. ábrán látható.



7. ábra. Optimális ablakméret meghatározásánál alkalmazott MLP neurális hálózat

Mivel a neuronoknál alkalmazott súlyok és eltolások kezdeti nagysága Gauss-eloszlásból származó véletlenszerű érték, ezért azonos adatokkal való többszöri tanítás esetén is kissé különböző osztályozási pontosságokat kapunk. Azért, hogy csökkentsem annak az esélyét, hogy emiatt téves következtetést vonjak le az ideális ablakméretre vonatkozóan, minden ablakméret mellet öt alkalommal ismételtem meg a kísérletet. A rendelkezésre álló adatok 70%-át tanításhoz, 30%-át teszteléshez használtam fel. A 4. táblázat tartalmazza a neurális hálózat osztályozásának pontosságát, azaz a helyes és az összes osztályozás számának arányát a tesztadatokon mérve, illetve a kapott pontosságok szórását.

4. táblázat. Neurális hálózat osztályozási pontossága különböző ablakméretek esetén

Ablakméret (mintaszám és időtartam)	**Pontosság a tesztadatokon**
40 (0,25 mp)	78,64 ± 1,30%
60 (0,475 mp)	86,90 ± 0,61%
80 (0,5 mp)	91,52 ± 0,69%
100 (0,625 mp)	93,76 ± 1,33%
120 (0,75 mp)	95,89 ± 0,15%
140 (0,875 mp)	97,57 ± 0,47%
160 (1 mp)	98,62 ± 0,21%

180 (1,125 mp)	98,34 ± 0,41%
200 (1,25 mp)	97,99 ± 0,33%

A kapott eredmények alapján az egy másodperc hosszú ablak használatát találtam optimálisnak ebben az alkalmazásban. Ez azt jelenti, hogy amikor megjelenik a képernyőn, vagy eltűnik onnan az objektum, egy másodperc elteltével nagy biztonsággal megállapítható, hogy mit lát az önkéntes, amely elfogadható mértékű késleltetést jelent akár valós idejű alkalmazásokban is, emellett ugyanakkor nagy felismerési pontosságot tesz lehetővé. Egy személy esetén a módszer igen nagy pontosságot biztosított, azaz az alkalmazott gépi tanulási lánc képes azokat a mintázatokat megtalálni az adatokban, amelyek alapján az egyes tevékenységek elkülöníthetők. Egy másik személy, illetve tevékenység esetén, ha ezek a mintázatok nem is azonosak, jellegükben hasonlóak, ugyanis más adatfájlokkal történő vizsgálataim is a fentebb vázolt megállapításokat támasztották alá, tehát az optimális szegmensméretre vonatkozó eredmény a többi esetre, azaz más önkéntesekre és más típusú tevékenységekre is általánosítható. Egyetlen személy adatain betanított tanulóalgoritmus ugyanakkor nem fog jól teljesíteni mások adatain, hiszen ebben az esetben nagyon sok egyénspecifikus mintázatot tanul meg a hálózat, így általánosítási képessége gyenge lesz.

Érdemes kitérni arra is, hogy az egy másodpercnél hosszabb ablakméretek esetén miért tapasztalható a pontosság kismértékű csökkenése, miközben a tanulóalgoritmus számára egy nagyobb méretű szegmens több információt biztosít. Az eredeti jel négy másodperces hosszúságú, és ekkora ablak esetén vizuálisan az amplitúdó ingadozása nem tűnik relevánsnak olyan mérőszámok kiszámításakor, mint például az átlag és a szórás. Azokat a jellemzőket, amelyek a legjobban leírhatnak egy felismerendő tevékenységet, a hálózat nem találja meg, mivel ezek az ingadozások kis súllyal szerepelnek

a hálózat rejtett rétegeiben. Ugyanakkor, ha az eredeti jelből egy rövid, 0,25 másodperces ablakot veszünk, az ingadozások relevánsabbak, és az átlagtól való minden eltérés nagyobb súlyt kap, de egyúttal ez nagyobb bizonytalanságot teremt, amikor a hálózatnak be kell sorolnia az ablakot egy bizonyos kategóriába. Az egy másodperces hosszúságú ablak megfelelőnek tűnik az osztályozáshoz, mert a jel ingadozásainak tulajdonított súly optimális, a jel bizonyos ingadozásai minden bizonnyal a kéz szorításából vagy a résztvevő személy által végrehajtott egyéb műveletekből származnak a mérésben, és ezeket az ingadozásokat a hálózat e cselekvések indikátoraként ismeri fel. Az itt ismertetett eredmények alapozzák meg a második tézispontot.

2. tézis: *Meghatároztam azt az optimális szegmensméretet, amellyel a gépi tanulási modell teljesítménye maximalizálható EEG-alapú tevékenységfelismerés esetén. Megállapítottam, hogy a szegmensméret növelésével a felismerési pontosság is szigorúan monoton növekvő egy másodperces méretig, ez után viszont a pontosság kis mértékű csökkenése következik be.*

4.2. Normalizálás

A fejbőrön mérhető EEG jelek nagysága alanyonként és mérésenként is változhat, amelynek oka többtényezős. Egyrészt szerepet játszik benne az izzadás, amely szignifikánsan csökkenti az elektróda-bőr közötti impedanciát [47]. Másrészt nemek közötti különbség is megfigyelhető: nők esetén nagyobb az EEG jelek amplitúdója, mint férfiaknál [48]. Szerepet játszik továbbá az öregedés is, hiszen az életkor előrehaladtával csökken a jelek nagysága [49]. Ennek a hatásnak a kiküszöbölésére normalizálást alkalmazhatunk, amely lehetővé teszi, hogy a különböző mérésekből, egyénektől, stb. származó adatok

dinamikai tartománya azonos legyen. A művelet elvégezhető közvetlenül a nyers adatokon, vagy a kinyert jellemzőkön is.

A normalizálás tulajdonképpen egy leképezés, az adatokat transzformálja egyik térből a másikba, amely többféle módon történhet. Az egyik ilyen technika a minimum-maximum normalizáció, ahol az adatokat a [0, 1] tartományba képezzük le, azaz az adatvektorban megtalálható legnagyobb értékhez az 1-et, legkisebbhez a 0-t, a köztes értekekhez pedig 0 és 1 közti számot rendelünk. Ha az adatvektort $\boldsymbol{f}$ jelöli, f_i pedig az adatvektor egy elemét, akkor az összes f_i elemre alkalmazva az (1) összefüggést megkapjuk a normalizált adatvektor elemeit.

$$f_i^{norm} = \frac{f_i - \min(\boldsymbol{f})}{\max(\boldsymbol{f}) - \min(\boldsymbol{f})} \tag{1}$$

Egy másik adatnormalizációs módszer a decimális skálázás, amely egy nemlináris leképezés. Ez néhány esetben (adattartalomtól függően) jobb teljesítmény nyújthat. Decimális skálázás esetén a (2) összefüggés használatával számíthatók a normalizált adatvektor elemei. Az összefüggésben j az eredeti $\boldsymbol{f}$ adatvektorban megtalálható legnagyobb érték számjegyeinek száma.

$$f_i^{norm} = \frac{f_i}{10^j} \tag{2}$$

Egy harmadik, szintén gyakran alkalmazott technika a zéruspont (*z-score*) vagy standard normalizálás. A normalizált értékeket a (3) egyenlet adja meg, amelyben μ_f az $\boldsymbol{f}$ adatvektor átlagát, σ_f pedig a szórását jelöli.

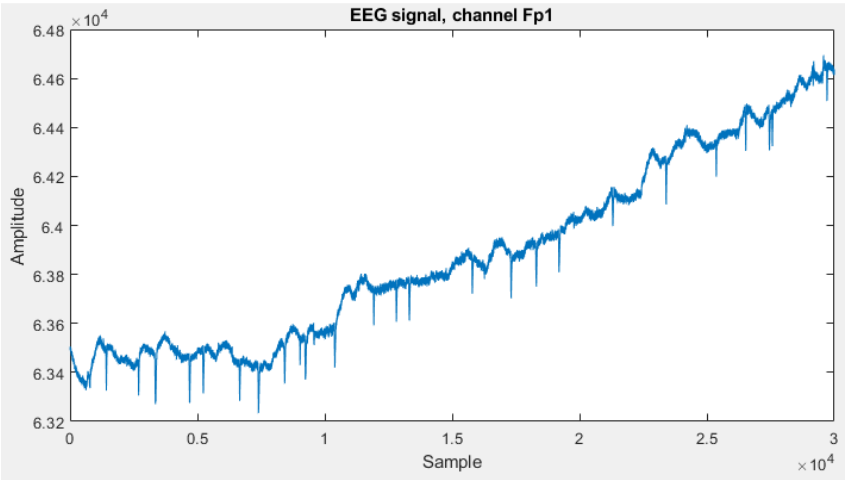
$$f_i^{norm} = \frac{f_i - \mu_f}{\sigma_f} \tag{3}$$

A kutatók a [50] cikkben az előbbi három módszert alkalmazták MLP hálózattal végrehajtott osztályozás esetén, és megmutatták, hogy ezek közül bármelyik megközelítés alkalmazása lényegesen javítani tudja a hálózat

hatékonyságát, ugyanakkor közöttük már nincs jelentős különbség, mindössze egyetlen mintánál tért el az eredmény a különböző normalizálási módok használatánál. Ennek okán a továbbiakban minimum-maximum normalizálást alkalmaztam.

4.3. Trendmentesítés

EEG jelek feldolgozása esetén gyakran felmerülő probléma akár a rossz jel-zaj viszony, akár az alapvonal nagyságához viszonyított relatív kicsi, valódi információt hordozó jel nagysága. A PhysioNet adatbázisban ugyan nem, de saját méréseim során gyűjtött adatokban ezt jelentős mértékben tapasztaltam. A 8. ábra egy két perc hosszúságú, önkéntes személyen végzett, az Fp1 csatornán általam rögzített adatsort ábrázol. A mérést a korábban említett Ultracortex eszközzel és saját fejlesztésű adatgyűjtő szoftverrel végeztem.



8. ábra. Saját mérésem során rögzített adatok az Fp1 csatornán

Az ábrán jól látszik két különböző probléma. Az egyik, hogy az alapvonal nagyságrendjéhez ($\sim 10^4$) képest az arra szuperponálódó, információkat hordozó jel nagysága igen kicsi ($\sim 10^2$), annak kb. 1%-a. Ez – különösen, ha valamilyen hardveres implementáció során a kisebb

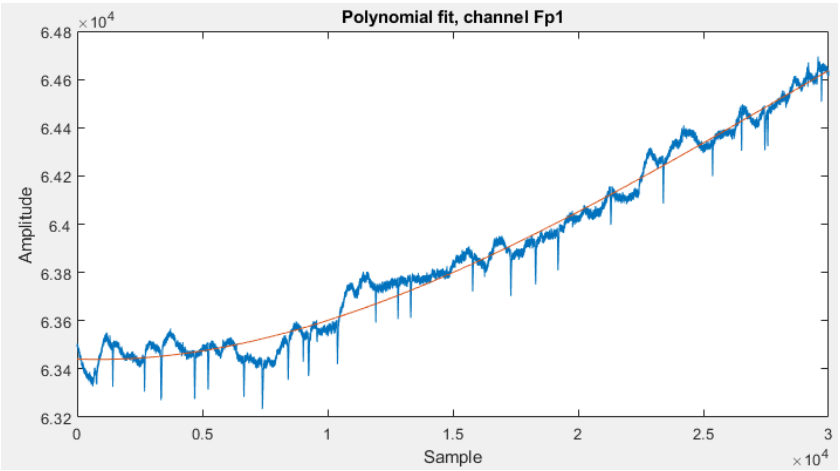
erőforrásigény érdekében alacsonyabb számábrázolási pontosságot alkalmazunk –, további problémákat okozhat.

A másik szembeötlő jelenség pedig az alapvonal nagyságának az idő telésével történő folyamatos növekedése. Az EEG hajlamos az alapvonal lassú eltolódására (drift), amely a bőr-elektróda interfészen keletkezik. Ezt a hatást alapvetően a mérés során az izzadásból származó elektrolit okozza. A fémelektródákkal reagáló elektrolit folyamatos amplitúdónövekedést idéz elő, amely szennyezi az adatokat, és megnehezíti azok elemzését és értelmezését [51]. Saját mérési adataimban minden csatornánál szignifikáns folyamatos amplitúdónövekedést tapasztaltam. Ezt trendnek nevezik: egy idősor szisztematikus változása, amely nem tűnik periodikusnak. Ezek a tendenciák elrejthetik a lassú, de valódi agykérgi aktivitást [52], ebből adódóan el kell őket távolítani az adatokból a következő lépések előkészítéseként.

A trend eltávolítására több különböző módszer alkalmas. A szokványos és elterjedt módszer egy felüláteresztő szűrő használata, hiszen ez képes a lassú, folytonos növekedés kiszűrésére, hátránya ugyanakkor, hogy érzékeny az időben lokális hibák jelenlétére, ugyanis ezek a kimenet oszcillációját okozzák [53].

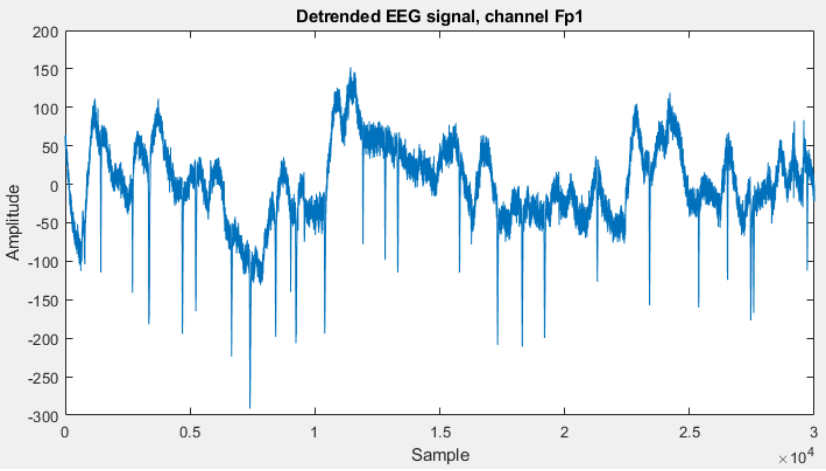
Egy másik, trendmentesítésre használható megoldás a függvényillesztés, amely során valamilyen függvényt, tipikusan polinomot illesztünk az idősor adataira, majd a függvényértéket kivonjuk az adatokból. A polinom fokának megválasztásával szabható meg, hogy milyen típusú tendencia (pl. elsőfokú esetén lineáris növekedés vagy csökkenés) szűrhető ki az adatokból. Túl alacsony fokú polinom használata esetén előfordulhat, hogy az nem követi az alapvonal változását megfelelően, míg túl magas fokúnál olyan helyi kilengéseket is lekövet az illesztett függvény, amelyeket valójában már az információt hordozó jel változása okoz, nem pedig az alapvonal folyamatos

növekedése. A fokszám meghatározására egy szisztematikus módszert alkalmaztam: egy nulladfokú vagy elsőfokú polinom egyértelműen nem képes lekövetni az alapvonal eltolódását, ez már az adatsor grafikus ábrázolását követően látható, minimum másodfokút kell számításba venni. Másodfokú polinom esetén azonban még mindig észrevehetők egészen alacsony frekvenciájú hullámzások az adatokban – a 120 másodperces mérésben 1-2 hullám, azaz század Hz nagyságrendű –, amelyekről nem valószínű, hogy az agy működésével kapcsolatban állnak, hiszen a legalacsonyabb frekvenciájú delta hullámsáv alsó határa 0,3-1 Hz körüli – bár nincs rá egzakt érték, de mindenképp legalább egy nagyságrenddel magasabb az itt tapasztalható frekvenciánál. Harmadfokú polinom esetén ez a frekvenciakomponens már teljesen eltűnik a jelekből. A negyedfokú még épp elfogadható, azonban tovább növelve a fokszámot már olyan csúcsokra is elkezd illeszkedni a függvény, amelyek minden bizonnyal valódi információt hordoznak. Ez alapján a mért adatokra egy harmadfokú polinom illesztése tűnt megfelelő választásnak, amelyet minden pontban kiértékeltem. Az első csatorna (Fp1) adataira illesztett görbe a 9. ábrán látható.



9. ábra. Mérési adatokra illesztett harmadfokú polinom

A függvényillesztést követő lépés a függvényértékek kivonása az adatsor pontjaiból. Ezzel olyan jelhez jutunk, amelynek értéke nulla körül fluktuál és nem figyelhető meg a mérés során tapasztalt folytonos növekedési tendencia. A trendmentesített adatsort a 10. ábra mutatja be.



10. ábra. Az Fp1 csatorna trendmentesített mérési adatai

4.4. Jellemzők kinyerése

Önmagában a mérési adatok használata a gépi tanulóalgoritmus bemeneteként jellemzően nem képes maximalizálni annak teljesítményét, szükséges lehet különböző jellemzők (leírók) kinyerése, illetve kiszámítása a nyers mérési adatokból. Ez a művelet szintén az adatok előfeldolgozási lépései közé sorolható és a kapott jellemzők a gépi tanulási modell bemeneteként felhasználhatók akár önmagukban, akár a mérési adatok kiegészítéseként, javítva ezzel a modell teljesítményét.

A jellemzők kinyerése történhet akár statikus módon, akár dinamikusan bizonyos típusú tanulóalgoritmusok (pl. CNN) esetén. Kutatásomban arra kerestem a választ, hogy MLP hálózat esetén milyen mértékben tud hozzájárulni a felismerési pontosság növeléséhez, ha a mérési adatok mellett

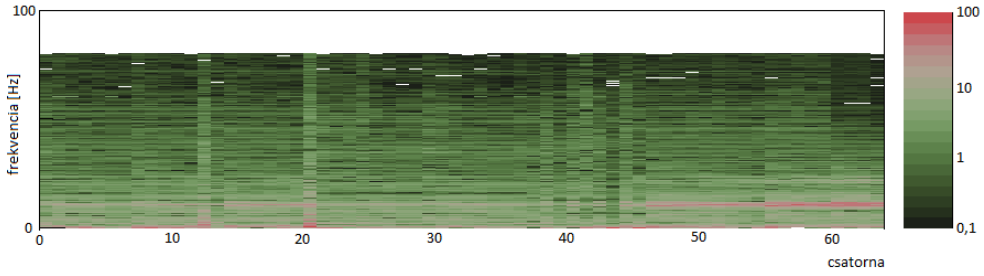
kinyert jellemzőket is használok a tanulóalgoritmus bemeneteként, illetve, ha csak és kizárólag a kinyert jellemzőket használom nyers mérési adatok nélkül.

4.4.1. Fourier-transzformáció

Az EEG-n az idegi oszcillációk a frekvenciatartományban figyelhetők meg, amelyeket általában Fourier-transzformáción alapuló spektrális módszerekkel nyernek ki. Mintavételezett adatok lévén a diszkrét idejű Fourier-transzformáció (4) alkalmazható, amely mérési adatok egy x sorozatára megadja annak spektrumát.

$$X(\omega) = \sum_{k=-\infty}^{\infty} x[k] e^{-i\omega k} \tag{4}$$

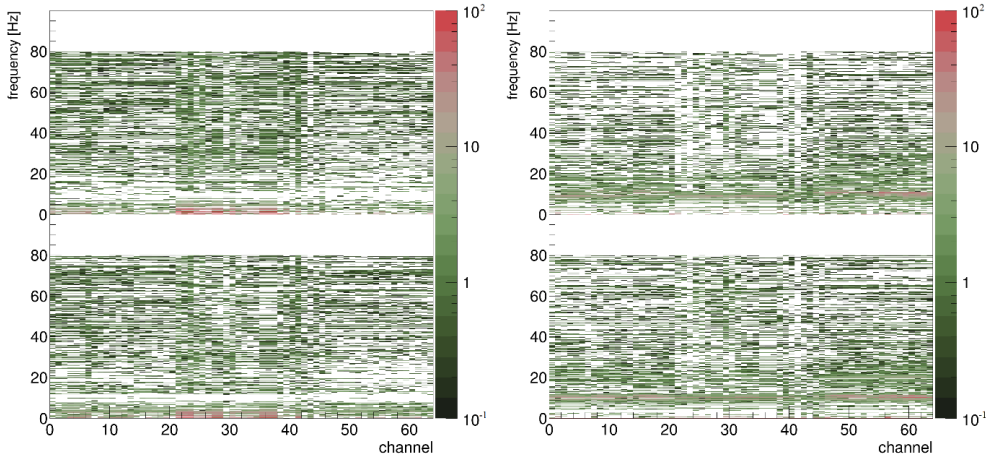
Munkám során többek között azt is vizsgáltam, hogy a szem csukott, illetve nyitott állapota esetén megfigyelhető-e (neurális hálózattal) valamilyen különbség az agyhullámokban. Ehhez a PhysioNet adatbázis releváns adatait és a Fourier-transzformációt használtam fel. A frekvenciatartományba történő konvertálást az adatok szegmentálását követően mind a 64 csatornán elvégeztem, majd az amplitúdókat a frekvencia és csatorna függvényében kétdimenziós hisztogramon ábrázoltam. A 11. ábra egy négy másodperces ablak alapján számított spektrumot mutat be.



11. ábra. Fourier-transzformáció eredménye

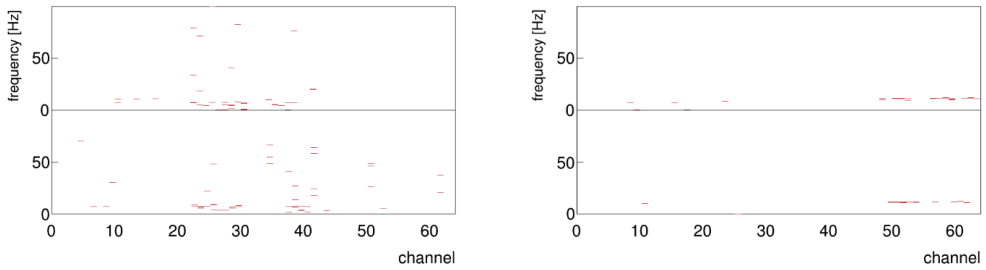
Annak érdekében, hogy a két különböző feladat végrehajtása során az agyi tevékenységben bekövetkező változásokat el lehessen különíteni az ettől

független, egyébként is jelen lévő agyi tevékenységtől, elkészítettem minden önkéntes esetén azokat a hisztogramokat, amelyek minden egyes csatorna és frekvencia esetén megmutatják az amplitúdók különbségét a zárt-nyitott és nyitott-zárt esetben is. Az egyik önkéntes első két időszelete esetén kapott eredmény a 12. ábrán látható.



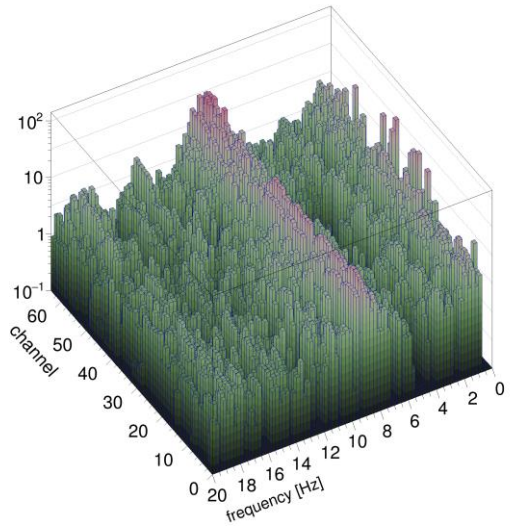
12. ábra. Különbségi hisztogramok nyitott (bal oldal) és zárt (jobb oldal) szem esetén

A 12. ábrán jól látható, hogy a zaj jellegű különbségekhez mérten bizonyos csatornák és frekvenciák esetén több nagyságrendbeli eltérés jelentkezik az amplitúdókban. Ennek alaposabb vizsgálatára Gauss-görbe illesztést használtam, amelyet csatornánként külön-külön végeztem el, az illesztési ablak frekvencia mentén történő folyamatos eltolásával, többféle ablakméret mellett. Abban az esetben, ha az illesztett függvény amplitúdója a vizsgált ablakban meghaladt egy bizonyos küszöbértéket, a várható értéke a 13. ábrán látható módon egy hisztogram megfelelő cellájában megjelenítésre került.



13. ábra. K=10 küszöbértéket meghaladó amplitúdójú függvények várható értéke nyitott (bal oldal) és zárt (jobb oldal) szem esetén 4 Hz illesztési ablakméret mellett

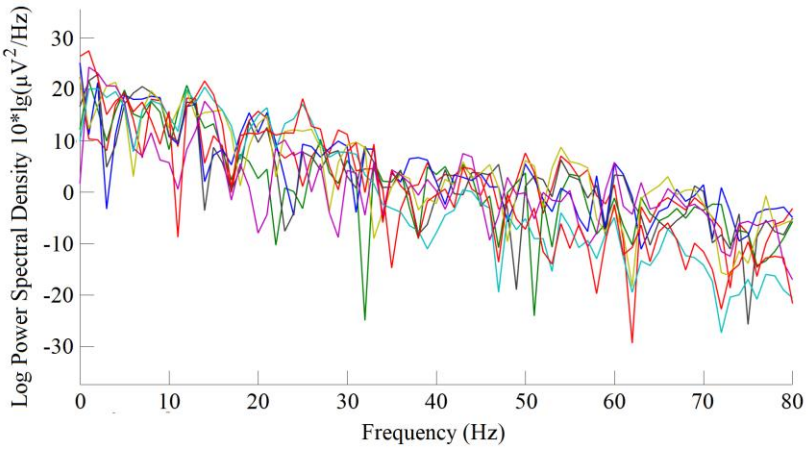
A függvényillesztés eredményéből látható, hogy elsősorban a magasabb sorszámú csatornák esetén 10 Hz-es frekvencia környékén fokozott agyi aktivitás mérhető. Az 2. ábrán látható számozás alapján ezek a csatornák (47-64) a nyakszirti lebeny területén találhatók. Ez a megfigyelés egyezik azon szakirodalmi adatokkal, mely szerint a nyakszirti területek fölötti magasabb amplitúdójú alfa-hullám leginkább a becsukott szemű, de éber nyugalmi állapottal korrelál [54]. Közelebbről megnézve, 3D-s ábrázolással jól látható a zárt szem esetén megjelenő csúcs a 14. ábrán.



14. ábra. Zárt szem esetén megjelenő alfa-hullám, a nyakszirti lebeny területén magasabb amplitúdóval

4.4.2. Teljesítménysűrűség-spektrum

A Fourier-transzformáción alapszik a teljesítménysűrűség-spektrum (PSD) is, amely a spektrális teljesítménysűrűséget adja meg, azaz a jel különböző frekvenciájú összetevői által hordozott teljesítményt. A PhysioNet adatbázis egyik önkéntesének méréseiből egy egy másodperc hosszúságú ablakából (160 minta) számított teljesítménysűrűség-spektrumot a 15. ábra mutatja be, amelyen a színek egy-egy csatornát reprezentálnak.



15. ábra. Teljesítménysűrűség-spektrum

A ritmikus agyi aktivitás frekvencia szerint sávokra bontható. Ezek megkülönböztethetők alfa, béta, gamma, delta és théta hullámokként. Ezen elnevezésekkel jelölt frekvenciatartományok nem egzakt módon definiáltak, különböző szerzők kismértékben eltérő értékeket használtak [55-57]. Az általam használt frekvenciasávok a következők voltak:

- alfa: 8-14 Hz
- béta: 14-30 Hz
- gamma: 30-80 Hz
- delta: 1-4 Hz
- théta: 4-8 Hz.

A jel abszolút teljesítménye az egyes hullámsávokban ezen tartományhatárok felhasználásával került kiszámításra.

4.4.3. Független komponens analízis

Bár EEG adatok rögzítése esetén az elektródák a fejbőr különböző területein helyezkednek el, ez nem jelenti azt, hogy kizárólag az onnan származó jeleket mérik, hiszen – ugyan alacsonyabb amplitúdóval –, de más területekről származó jelek is elérik az elektródákat, így azok valójában egy keveréket érzékelnek. Ezen kívül egyéb, pl. szemmozgásból vagy pislogásból származó zajok szintén rögzítésre kerülnek.

A jelfeldolgozásban a független komponens analízis (*independent component analysis*, ICA) egy módszer a többváltozós jelek összeadódó részkomponensekre történő szétválasztására. A módszer lényege, hogy a rendelkezésre álló adatok megadhatók ismeretlen, de statisztikailag független részkomponensek (jelek) lineáris kombinációjaként. A feladat ezen ismeretlen részkomponensek meghatározása, ilyen módon a szenzorok által érzékelt kevert jelek szétválaszthatók a független forrásokra. Egy összetett, kevert jel független komponensekre történő felbontása megkönnyítheti a későbbi feldolgozási lépéseket, ideértve a gépi tanulási módszerrel történő osztályozást. Ez alapján a független komponens analízist is jellemzőkinyerési eljárásnak tekinthetjük.

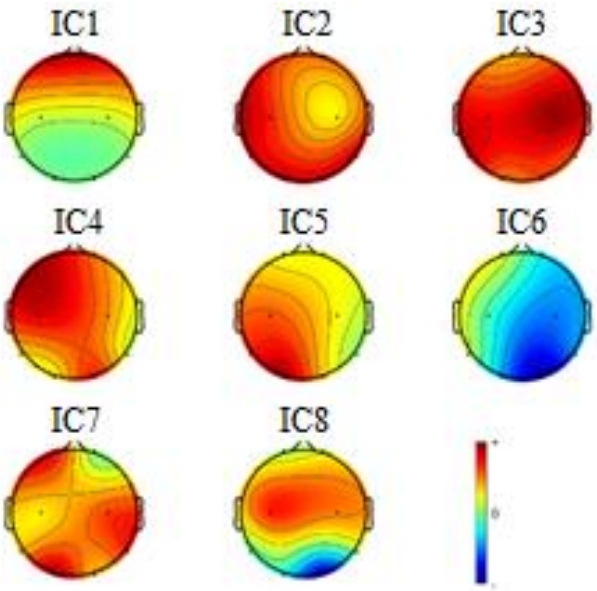
A mért adatok leírhatók független források (komponensek) és a keverési információ lineáris kombinációjaként az (5) összefüggés segítségével:

$$x_t = As_t \tag{5}$$

ahol x_t a megfigyelt jelek vektora, s_t az eredeti forrásjelek vektora és A a keverő mátrix. Az eredeti, független források pedig meghatározhatók a (6) egyenlet felhasználásával:

$$\widehat{s_t} = W x_t \tag{6}$$

ahol $W = A^{-1}$ a feloldási mátrix. Ennek kiszámításához az Infomax algoritmust használtam, amely az entrópia maximalizálásán alapul [58]. Az ICA dekompozíció eredményét a PhysioNet adatbázis egyik mérési fájljának adatain a 16. ábra szemlélteti.



16. ábra. Független komponens analízis eredménye

A független komponensek típusainak megfelelő felismerése tapasztalatot igényel. Elmondható azonban, hogy helye alapján az IC1 nagy valószínűséggel szemmozgásból, illetve pislogásból eredő komponens.

4.4.4. Eredmények értékelése

A független komponens analízis és a PSD osztályozási pontosságra gyakorolt hatásának vizsgálatához MLP neurális hálózat tanítását végeztem el a kapott adatokkal. A hálózat – az eredmények összehasonlíthatósága érdekében – azonos volt az ideális ablakméret meghatározásánál alkalmazottal,

azaz MLP két rejtett réteggel, rejtett rétegenként 20 neuronnal és szigmoid, a kimeneti rétegben pedig lineáris aktivációs függvénnyel.

Kezdetben egy darab PhysioNet adatfájl használatával vizsgáltam meg a PSD hatását. Az első esetben a neurális hálózat nyolc bemenettel rendelkezett, ahová a nyolc csatornáról származó nyers adatok érkeztek, azaz kinyert jellemzők itt nem kerültek felhasználásra. Az így kapott osztályozási pontosság 56,2%-ra adódott. Második esetben PSD használatával (0,5 másodperces ablakmérettel) meghatároztam az alfa, béta, gamma, delta és théta hullámsávban kapott abszolút teljesítményt mind a nyolc csatorna esetén és a nyers adatok mellett ezt is felhasználtam bemenő adatként, így 48-ra növelve a hálózat bemeneteinek számát. Az újratanítást követő tesztelés során 91,5%-os osztályozási pontosságot kaptam, amely jelentősen felülmúlta az előbbi, csak nyers adatokat használó hálózat teljesítményét. Ezek után megvizsgáltam, hogy milyen eredmény érhető el, ha kizárólag a kinyert jellemzőket használom fel nyers adatok nélkül. Azt tapasztaltam, hogy a pontosság nem csökkent ebben az esetben, viszont a tanítási idő a kevesebb bemenetnek köszönhetően – a várakozásoknak megfelelően – igen. Az előbbiek alapján konklúzióként megállapítható, hogy EEG adatok használata esetén jelentős teljesítményjavulás érhető el, ha a nyers adatok mellett vagy helyett a hullámsávokban mért teljesítményt, mint kinyert jellemzőt használjuk fel a tanulóalgoritmus bemeneteként. Ezen eredmények alapján fogalmazódott meg a harmadik tézispont.

3. tézis: *Megmutattam, hogy EEG adatok használata esetén jelentős pontosságnövekedés érhető el, ha a nyers adatok mellett vagy helyett a hullámsávokban mért teljesítményt, mint kinyert jellemzőt használjuk fel MLP tanulóalgoritmus bemeneteként. Az EEG jel abszolút teljesítményét kiszámítva az alfa, béta, gamma, delta és théta hullámsávokban, és ezeket használva*

bemenő adatként 91,5%-os, míg a nyers adatok közvetlen felhasználásával 56,2%-os felismerési pontosságot ért el ugyanaz a hálózat.

Ezt követően nagyobb mennyiségű, 50 fő mérési eredményeit tartalmazó adat használatával és leave-one-out módszerrel történő tanulási és tesztelési adathalmazra bontást követően megvizsgáltam, hogy ha nem a nyers adatok alapján történik a PSD számítása, hanem az ICA-ból származó aktivációs mátrix adatainak használatával, az jelent-e a hálózat teljesítménye szempontjából pozitív változást. A leave-one-out módszer lényege, hogy csak 49 alany adatait használtam fel a tanításhoz, az 50. önkéntes adatai kizárólag a kiértékelésben vettek részt. Ennek az az előnye, hogy olyan személytől származó tesztadatokkal dolgozik, akinek az adatai egyáltalán nem kerültek felhasználásra a tanításhoz, ilyen módon következtetések vonhatók le arra vonatkozóan, hogy a tanított neurális hálózat mennyire általánosan használható új, ismeretlen bemenő adatok esetén. A kapott eredményeket az 5. táblázat foglalja össze.

5. táblázat. ICA és PSD hatása az MLP hálózat teljesítményére

Használt adatok	**MSE a tanítási adatokon**	**MSE a tesztadatokon**
50 önkéntes, leave-one-out módszer, Bemenet: PSD a nyers adatokon	0,195	0,207
50 önkéntes, leave-one-out módszer, Bemenet: aktivációs mátrix	0,206	0,204

50 önkéntes, leave-one-out módszer, Bemenet: PSD az aktivációs mátrixon	0,197	0,199

A táblázat adataiból látható, hogy – együttesen nézve a tanítási és tesztadatokon elért súlyozott eredményt – önmagában az ICA alkalmazása rosszabb eredményt ad a nyers adatokon vett PSD-hez képest, az ICA és PSD együttes alkalmazása pedig hasonlót, mint a PSD önmagában. Ezek alapján a független komponens analízis elvégzését nem tartom szükségesnek. Mint korábban említettem, a szenzorok által érzékelt jel lineáris kombinációja az agy különböző területeiről származó jeleknek, amelyeket a független komponens analízis szétválaszt. Ugyanakkor egy neuron tulajdonképpen a bemenő jelek lineáris kombinációját képezi, ezáltal ismét összekeverve azokat, így a két folyamat ebből a szempontból egymás inverzének tekinthető. Bár elméletileg a tisztább jelek megkönnyíthetik a neurális hálózat tanulását, azonban ebben az alkalmazásban a végeredmény szempontjából nem hoz előnyt az ICA előzetes elvégzése.

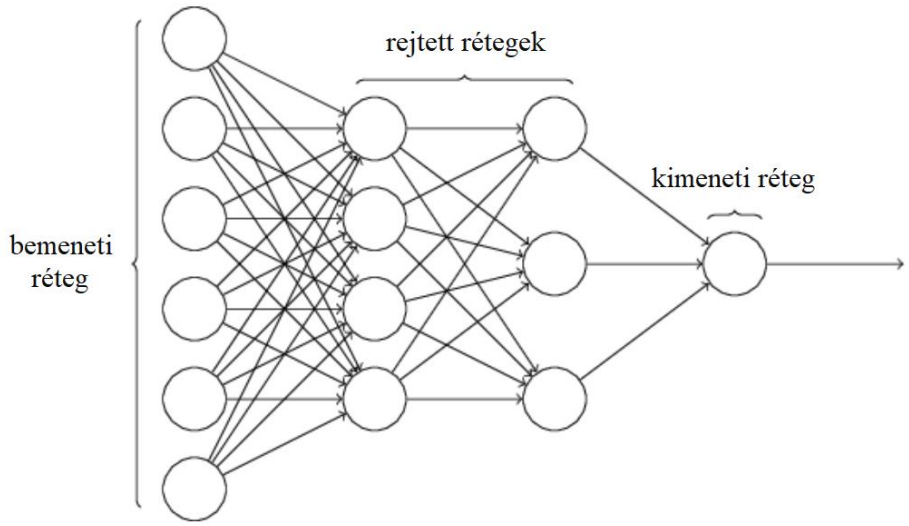
5. Neurális hálózatok felépítése

Ahogy azt a 2. fejezetben már részletesebben kifejtettem, EEG adatokból történő tevékenységfelismeréskor az optimális tanulóalgoritmus kiválasztása, illetve annak megfelelő paraméterezése közel sem egyértelmű, a különböző hivatkozott kutatások sok esetben egymásnak ellentmondó következtetésre jutottak. Mindemellett ugyanakkor megfigyelhető az a trend, hogy az ilyen jellegű tevékenységfelismerési feladatokban alkalmazott módszerek között egyre nagyobb teret hódítanak a konvolúciós neurális hálózatok.

Kutatásomban több eltérő MLP és CNN architektúrát használtam különböző forrásokból származó adatokon. Ebben a fejezetben egy rövid áttekintést kívánok adni ezeknek a hálózatoknak az általános felépítéséről, illetve azokról a paraméterekről, amelyeknek megfelelő megválasztása a tervező feladata a minél nagyobb hatékonyság elérése érdekében.

5.1. Többrétegű perceptron (MLP) hálózat felépítése

Egy MLP neurális hálózat felépítését tekintve egy előrecsatolt hálózat, amely egy bemeneti rétegből, egy vagy több rejtett rétegből és egy kimeneti rétegből áll. A bemeneti rétegbe érkeznek az adatok, ideértve a kinyert jellemzőket is, így az itt található neuronok száma a bemenő jelek számával (pl. ahány EEG csatorna van) egyezik meg. Innen az adatok továbbítódnak az első rejtett réteghez. A két réteg teljesen összekapcsolt, azaz a bemeneti réteg összes neuronja kapcsolódik az első rejtett réteg összes neuronjához. A további rétegek megelőző réteggel való kapcsolata szintén így adódik. A hálózat felépítését a 17. ábra szemlélteti.



17. ábra. Többrétegű perceptron hálózat felépítése

A rejtett rétegek és a kimeneti réteg minden neuronja tartalmaz egy súlyvektort, amely elemeinek száma megegyezik az előző réteg neuronjainak számával, egy eltolási értéket, illetve egy aktivációs függvényt. Egy neuron matematikailag a (7) alakban írható le:

$$f(p) = \varphi(\sum_{i=1}^{m} w_i \cdot p_i + b) \tag{7}$$

ahol $p \in \mathbb{R}^m$ a bemeneti vektor, $w \in \mathbb{R}^m$ a súlyvektor, $b \in \mathbb{R}$ az eltolás, $\varphi: \mathbb{R} \to \mathbb{R}$ pedig az aktivációs függvény.

A súlyok és az eltolási értékek kiszámítása a tanítás során automatikusan történik, azonban az aktivációs függvény, a rétegszám és a rétegenkénti neuronok számának meghatározása a tervező feladata. A neurális hálózatok szakirodalmát áttekintve számos javaslatot és útmutatást kaphatunk arra vonatkozóan, hogy egy MLP hálózat esetén hogyan érdemes ezeket megválasztani. A konklúzió azonban az, hogy kísérleti úton kell kitapasztalni mi működik a legjobban, ugyanis nem létezik jelenleg egy olyan eljárás, algoritmus, vagy módszer, amely ezt képes lenne egyértelműen megadni.

5.2. Konvolúciós neurális hálózat (CNN) felépítése

A CNN hálózatok esetén felmerülnek ugyanazok a kérdések, mint az MLP-nél, hiszen legtöbbször azok esetében is van egy MLP-jellegű rész a hálózatban (teljesen összekapcsolt rétegek), de emellett más típusú rétegek is találhatók bennük, tovább bonyolítva a feladatot ezen rétegek típusainak, paramétereinek meghatározásával. A CNN-alapú osztályozók két fő részre oszthatók. Az első rész végzi a jellemzők kinyerését, míg a második rész a jellemzők alapján az osztályozást.

5.2.1. Konvolúciós réteg

A konvolúciós réteg a jellemzők kinyerésében vesz részt, feladata a konvolúció, mint matematikai művelet elvégzése az adatokon. Ha f és g integrálható, $(-\infty, \infty)$ tartományban értelmezett függvények, akkor konvolúciójuk a (8) integrállal definiált függvény.

$$(f * g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dt \tag{8}$$

Mivel az adatok digitális formában állnak rendelkezésre, ezért esetünkben a konvolúció diszkrét idejű formája, azaz konvolúciós összeg alkalmazandó. Emellett a konvolúciós neurális hálózatok tipikusan képi adatok feldolgozására alkalmasak, amelyek kétdimenziós adatstruktúrák. Természetesen más jellegű bemenet is értelmezhető egyfajta képként, pl. az EEG adatsor, ahol a „kép” sorait az egyes csatornák, oszlopait pedig a mérési adatok adják. Az adatok kétdimenziós voltából fakadóan kétdimenziós konvolúció alkalmazására van szükség, amelynek diszkrét idejű formája:

$$y[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m,n]x[i-m, j-n] \tag{9}$$

ahol x a bemenő adatmátrix, h pedig a konvolúciós kernel, amelynek mérete az egyik olyan paraméter, amelyet a neurális hálózat tervezője határoz meg.

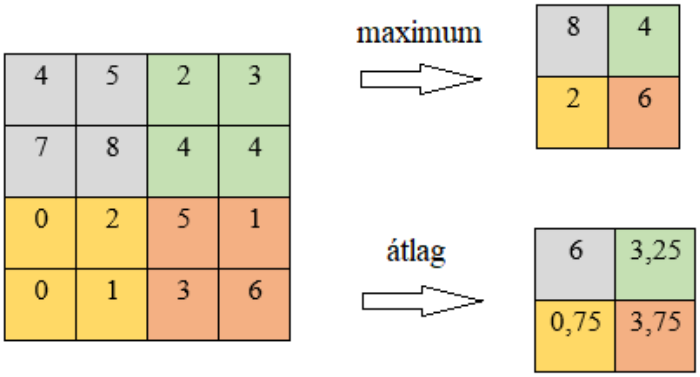
Belátható, hogy a konvolúció végrehajtása a kimenő adatmátrix méretének csökkenését okozza és a hatás a konvolúciós rétegek számának növelésével egyre fokozódik, amelyet tovább tetéz, ha a kernel léptetése nem egyesével történik, hanem valamilyen nagyobb lépésközzel, amely lépésköz megválasztása szintén a neurális hálózat tervezőjének feladata.

A kimeneti mátrix méretének folyamatos zsugorodását megelőzendő a bemenő adatmátrix minden oldalát további sorokkal és oszlopokkal kiegészítve, annak mérete megnövelhető olyan mértékben, hogy a konvolúciós kernel középső eleme az eredeti, nem megnövelt méretű mátrix bal felső elemére essen. Ez a párnázási (*padding*) módszer nulla értékek hozzáadását jelenti a mátrixhoz szimmetrikus módon.

Amint az az előbbiekből látható, konvolúciós réteg használata esetén a tervező feladata számos hiperparaméter meghatározása, ideértve a konvolúciós kernel méretét, a kernelek számát (amely a kimenet dimenzióját is meghatározza), a lépésközt és a párnázást.

5.2.2. Összevonó (pooling) réteg

Egy másik, konvolúciós neurális hálózatok esetén alkalmazott elem az összevonó (pooling) réteg, amely gyakran a konvolúciós rétegek között helyezkedik el. Ez a réteg az alulmintavételezés módszerének felhasználásával csökkenti a reprezentáció méretét, amellyel szintén csökken a paraméterek száma és így a számítások mennyisége a hálózatban. Összevonó réteg esetén gyakran 2x2 méretű szűrőket (ablakokat) alkalmaznak 2-es lépésközzel, amely 25%-ára redukálja a kimeneti mátrix méretét. Az egyes ablakokban számolható az átlagos, vagy maximális érték, amint az a 21. ábrán látható.



21. ábra. Összevonó réteg hatása

Maximum pooling esetén az eredmény úgy interpretálható, hogy az ablakon belül megtalálható-e az adott jellemző, míg az átlag kiszámítása esetén úgy, hogy átlagosan mennyire van jelen. Régebben jellemzően az utóbbi módszert alkalmazták, azonban a kutatások rámutattak, hogy a maximum pooling alkalmazásával jobb eredmény érhető el a gyakorlatban [59].

Összegezve elmondható az összevonó rétegről, hogy olyan paramétert nem visz a hálózatba, amelyet a tanítás során kellene kiszámítani, ugyanakkor a tervező feladata annak eldöntése, hogy milyen legyen az alkalmazott függvény, az ablakméret, a lépésköz, és az esetleges párnázás, így további hiperparaméterek meghatározása válik szükségessé.

5.2.3. További rétegek

Az előbbieken túl számos más típusú réteg is alkalmazható egy konvolúciós neurális hálózatban. Ezek a hiperparaméterek alacsony száma vagy akár teljes hiánya miatt tervezői szempontból egyszerűbben használhatók. Ezen rétegek egyike a batch normalizációs réteg. Egy neurális hálózat mélyítésével egyre komplexebb jellemzők megtanulására alkalmas eszközt kapunk, azonban sok rétegből álló hálózatoknál felmerülnek olyan

problémák, mint például a gradiensek eltűnése vagy túlfutása (robbanása). A gradiens eltűnés valószínűsége jelentősen megnő szigmoid aktivációs függvény alkalmazásakor, ugyanis a független változó értékének növelésekor a szigmoid deriváltja nullához tart. Mivel sztochasztikus gradiens tanítási módszer alkalmazásánál a parciális deriváltakat láncszabállyal számoljuk, ezért az ezzel számított szorzat is közel nulla lesz, a gradiens eltűnik, a konvergencia lelassul. A probléma megoldásának egyik lehetséges módja a batch normalizálás alkalmazása, amely úgy módosítja az aktivációs függvény független változójának értékét, hogy ne legyen a kapott gradiens nulla közeli. Ez a gyakorlatban egy normalizáló réteget jelent olyan módon, hogy az átlagérték nullának, a szórás pedig egynek adódjon a következő réteg számára.

A teljesen összekapcsolt rétegek a kinyert jellemzők alapján történő osztályozást végzik, felépítésük azonos az MLP hálózat rejtett rétegeivel, így a tervező által megválasztandó hiperparaméterek is azonosak, mint az 5.1. fejezetben említettek, azaz a neuronok száma a rétegben és az aktivációs függvény.

Mivel a teljesen összekapcsolt réteg számára egydimenziós adatvektorra van szükség, ezért az előző rétegekből származó többdimenziós adatokat egy speciális sorosító (*flatten*) réteg segítségével tudjuk átalakítani. Itt sem tanítható, sem pedig hiperparaméterek nincsenek jelen.

Néhány szoftverkörnyezet az aktivációs függvényt is egy külön rétegként kezeli (pl. Matlab), míg mások nem. Akadnak olyan rendszerek is (pl. Keras), amelyek mindkét megközelítést támogatják, azaz az aktivációs függvény lehet egy réteg része, de különálló rétegként is hozzáadható a hálózathoz.

6. Neurális hálózatok hatékonyságvizsgálata

Az előző fejezetben leírtak alapján látható, hogy egy CNN lényegesen komplexebb, mint egy MLP hálózat, több különböző réteg alkalmazható, sokféle kombinációban, a rétegeken belül pedig szintén számos paraméter megválasztása a tervező feladata. Ez egyfelől lényegesen nagyobb flexibilitást biztosít, mintha kizárólag a rétegek száma, mérete, és az aktivációs függvény típusa lenne megválasztható, mint az MLP esetén, ugyanakkor nehezebbé is teszi az ideális architektúra megválasztását.

Kutatásomban több különböző hálózat hatékonyságvizsgálatát végeztem el. Egyrészt kíváncsi voltam arra, milyen struktúrájú hálózattal érhető el a legnagyobb pontosság EEG adatok felismerése esetén, illetve hogy egy optimális architektúrával, és a korábbiakban említett különböző előfeldolgozási és jellemzőkinyerési módszerek felhasználásával túl lehet-e szárnyalni MLP hálózattal az automatikus jellemzőkinyerést végző CNN-ek teljesítményét.

6.1. Aktivációs függvény megválasztása

A neuronok kimenetén alkalmazott aktivációs függvény többféle típusú lehet, és gyakran a köztes, rejtett rétegekben, illetve a kimeneti rétegben használt függvény eltérő. A téma jelenlegi állása szerint MLP hálózat esetén gyakran alkalmazzák lineáris (10), a Gauss (11), a tangens hiperbolikusz (tanh) (12), és a szigmoid (13) típusú aktivációs függvényeket. Természetesen ezen kívül más típusú függvények is használhatók, mint pl. a ReLU (rektifikált lineáris egység, rectified linear unit), amely a pozitív értékeket nem módosítja, a negatívakhoz viszont egységesen nullát rendel (14). Ez utóbbit azonban az

általam használt ROOT környezet nem támogatta, így méréseket csak a másik négy típussal végeztem el.

$$\varphi(x) = x \tag{10}$$

$$\varphi(x) = e^{-x^2} \tag{11}$$

$$\varphi(x) = tan\, h(x) = \frac{2}{1+e^{-2x}} - 1 \tag{12}$$

$$\varphi(x) = \frac{1}{1+e^{-x}} \tag{13}$$

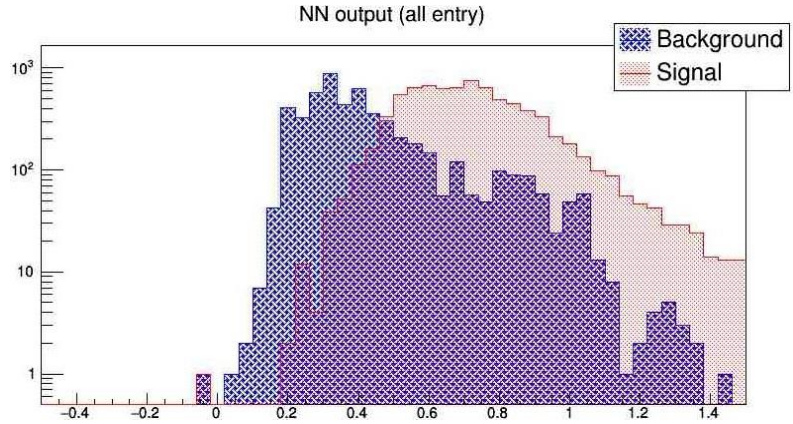
$$\varphi(x) = \max(0, x) \tag{14}$$

A tangens hiperbolikusz és a szigmoid függvények előnye, hogy korlátosak, azaz a következő réteg által kapott értékek a [-1;1] (tanh) vagy a [0;1] (szigmoid) intervallumba esnek.

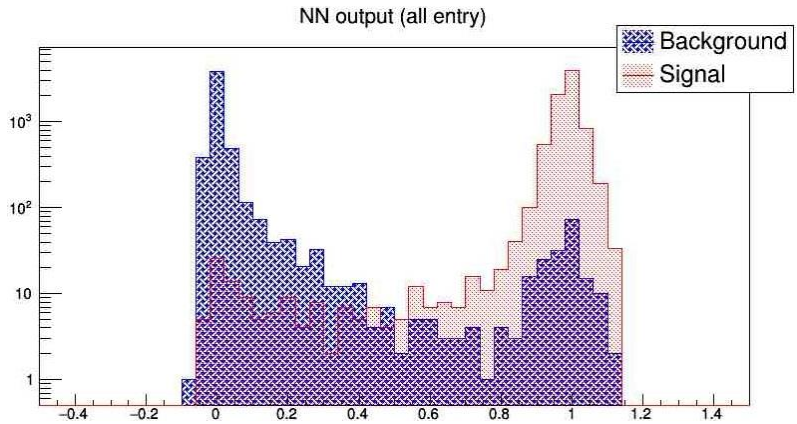
Az említett aktivációs függvények összehasonlítására egy két rejtett réteggel rendelkező MLP hálózatot használtam, rétegenként 5-5 neuronnal. A kimeneti réteg aktivációs függvénye minden esetben lineáris volt, a rejtett rétegeké pedig a felsoroltak egyike. A bemenő adatok ugyan nem EEG jelek voltak ebben az esetben, hanem egy gáztöltésű müondetektor jelei, azonban a korábbi eredmények azt mutatták, hogy ha osztályozási problémáról van szó, akkor jellemzően egy adott típus működik jól, függetlenül attól, hogy milyen jellegűek az adatok. Ezt a megállapítást alátámasztja a [60] cikk is: a kutatók tíz különböző jellegű adatkészleten vizsgálták pontosan annak a négy aktivációs függvénynek a hatását, amelyet én is használtam. Az elért eredményeik azt mutatták, hogy a szigmoid teljesítménye felülmúlja a másik három függvényét, függetlenül a használt adatkészlettől.

A müondetektor minden egyes csatornáján, ha a jel egy adott küszöbérték fölött volt, akkor a neurális hálózat az adott bemeneten 1-et, ellenkező esetben 0-t kapott, így 64 csatornás berendezésről lévén szó, 64

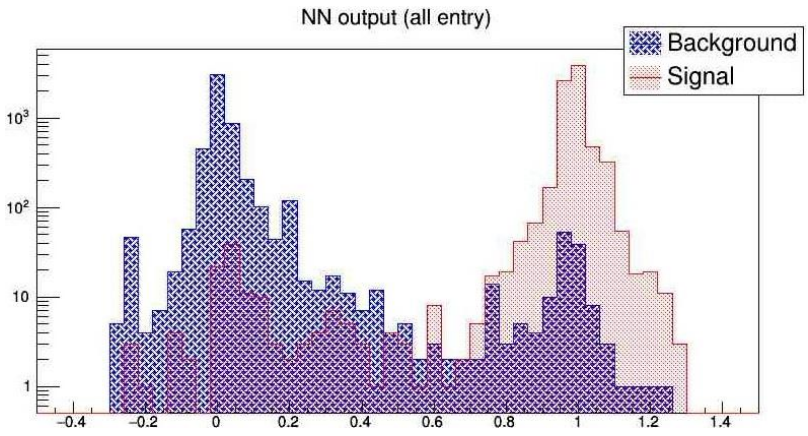
bemenetű hálózatot alkalmaztam. A címkézés pedig jel vagy háttér, attól függően, hogy ténylegesen érkezett-e müon a detektorba vagy sem az adott pillanatban. A különböző aktivációs függvényekkel elért eredményeket a 22-25. ábra mutatja, vízszintes tengelyen a hálózat kimenetének értéke, függőleges tengelyen pedig az eseményszám található logaritmikus skálán.



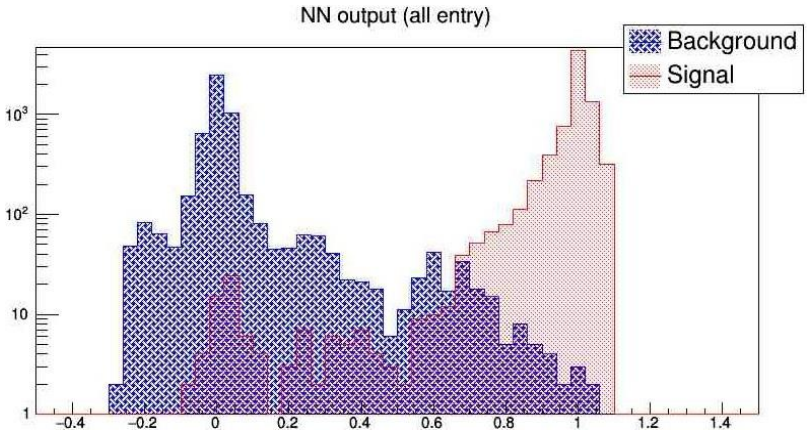
22. ábra. Neurális hálózat kimenete lineáris aktivációs függvény használatakor



23. ábra. Neurális hálózat kimenete Gauss aktivációs függvény használatakor



24. ábra. Neurális hálózat kimenete tanh aktivációs függvény használatakor



25. ábra. Neurális hálózat kimenete szigmoid aktivációs függvény használatakor

Az előbbi ábrákon látható, hogy szigmoid függvény használatakor a neurális hálózat (a kimenetre pl. 0,5 küszöbértéket alkalmazva) nagyobb hatékonysággal tudja megkülönböztetni a két osztályt, mint a többi alkalmazott aktivációs függvény esetén. Vizuális kiértékelés alapján a gyengébb eredmény ugyan elsősorban a lineáris függvény esetén látványos, azonban a numerikus adatok a többi függvény esetén is alátámasztják azok szigmoiddal szembeni alacsonyabb hatékonyságát.

6.2. MLP hatékonyságvizsgálata PhysioNet adatokon

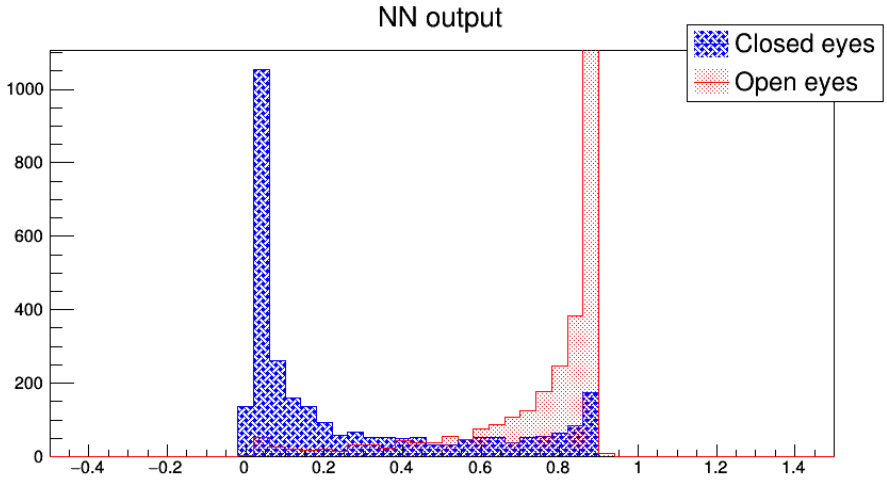
A PhysioNet adatbázison különböző architektúrájú MLP és CNN hálózatokat hasonlítottam össze annak érdekében, hogy megvizsgáljam, hogy EEG adatok felismerése esetén milyen teljesítményt nyújtanak, illetve a korábban kifejtésre került adat-előfeldolgozási módszerek alkalmazásával milyen mértékben javítható ez.

A tanítás során a Levenberg-Marquardt hibavisszaterjesztési algoritmust alkalmaztam, ugyanis ezzel tízszer-százszor nagyobb sebesség érhető el, mint a szokásos gradiens módszerrel [61]. A kezdeti súlyok és eltolási értékek Gauss-eloszlásból származtak. A mintahalmaz végigfuttatásának felső korlátjaként (epoch limit) 1000 iterációt alkalmaztam.

MLP esetén kétféle típusú adatfájlon dolgoztam, ezek közül az egyik a szem csukott és nyitott állapotának megkülönböztetésére szolgált. Mint azt a Fourier-transzformációval foglalkozó fejezetben kifejtettem, a nyakszirti lebeny területén található EEG csatornák (47-64) adatai frekvenciatartományba történő transzformálást követően jól használhatók a szem állapotainak megkülönböztetésére. Az így kapott adatok felbontása túl finom volt, így a teljes alfa hullámtartományban integráltam azokat és ezek az integrálok szolgáltak a neurális hálózat bemeneteként. A tanítás során az adatbázis első 50 önkéntesének adatait használtam fel.

Az osztályozást egy három rejtett rétegből, rétegenként három neuronból álló hálózat végezte, amelyek esetén – az előző pontban leírt tapasztalatok következményeként – szigmoid aktivációs függvényt alkalmaztam. A hálózat kimenetének értékét a 26. ábra mutatja be. Az ábrán látható, hogy a szem két állapotát jól meg tudja különböztetni a hálózat. A kimenetre 0,5-ös vágást alkalmazva az osztályozási pontosság 82,3%-ra adódik. Kipróbáltam több rétegből és több neuronból álló hálózatokat is, azonban – feltehetően a

probléma egyszerűségéből fakadóan – nem adódtak ezekkel sem jobb eredmények.



26. ábra. Szem állapotainak megkülönböztetése

A másik típusú adatfájl ettől komplexebb tevékenységről tartalmazott adatokat, a kézfej és a lábfej elképzelt mozgatásáról, illetve a pihenési állapotról. A várható összetettebb mintázatok miatt ebben az esetben 20-20, illetve 60-60 darab neuront használtam a rejtett rétegekben, amelyeket itt is szigmoid aktivációs függvénnyel láttam el. A kimeneti rétegben három neuron található, a három osztálynak megfelelően (kézfej használata, lábfej használata, pihenés), lineáris aktivációs függvénnyel. Ennek oka az, hogy ezt a hálózatot Matlab segítségével implementáltam, amely a kimeneti réteg esetén más megközelítést alkalmaz, mint a ROOT: minden osztályhoz egy-egy külön neuront rendel.

Kezdetben azt a neurális hálózatot használtam, amely 20-20 neuront tartalmazott a rejtett rétegekben. A korábbi fejezetekben és az 5. táblázatban összefoglalt eredményeket szintén ezzel értem el. Annak ellenére, hogy a kapott MSE értékek elfogadhatóak voltak, maga az osztályozási pontosság, azaz a helyes osztályozások aránya az összeshez viszonyítva, nem bizonyult

kielégítőnek. Ennek egyik lehetséges okának az adatkészlet kiegyensúlyozatlanságát véltem. A kiegyensúlyozatlanság forrása, hogy minden négy másodperces aktív tevékenységet (kézfej, illetve lábfej használata) egy szintén négy másodperces pihenés követett, azaz a pihenés során rögzített adatok mennyisége kétszeres volt a többi osztály adataihoz képest. Ennek kiküszöbölése érdekében véletlenszerűen kiválasztásra és eltávolításra kerültek az adathalmazból pihenés idején rögzített mérési adatok, úgy, hogy minden osztály mintái azonos arányban legyenek jelen.

Az adathalmaz kiegyensúlyozása önmagában nem hozott szignifikáns javulást a pontosság tekintetében, így megnöveltem a neuronok számát a rejtett rétegekben 60-60 darabra. A hálózat tanításához 10 fő adatait használtam fel, az adatok 70%-30% arányban lettek tanítási és tesztelési halmazra bontva, véletlenszerűen. A tanítási folyamat az általam használt Intel Core i7-4790 processzort és 8 GB RAM-ot tartalmazó számítógépen körülbelül 100 órát vett igénybe. Bár nem a nyers adatokat használtam fel, hanem a PSD-ből származó, öt hullámsávban kapott abszolút teljesítményt, amelyről korábban megállapítottam, hogy jelentős javulást hoz a nyers adatok közvetlen felhasználásához képest, azonban a tesztadatokon mérve így is csupán 71,5%-os osztályozási pontosságot sikerült elérni, amelyet nem tartottam kielégítőnek.

6.3. CNN hatékonyságvizsgálata PhysioNet adatokon

Összehasonlításképpen megvizsgáltam, hogy milyen eredményeket tudok elérni ugyanezen az adathalmazon CNN használatával. A CNN automatikus jellemzőkinyerési lehetőségét kihasználva a szegmentáláson kívül más előfeldolgozást nem végeztem. A szegmentálás során az adatokat 8x64 méretű blokkokra bontottam, ezek szolgáltak a hálózat bemeneteként. A mátrix

nyolc sora a nyolc csatornát, a hatvannégy oszlopa pedig ennyi mérési pontot (azaz 0,4 másodpercet) reprezentál. Azok a szegmensek eldobásra kerültek, ahol a 0,4 másodpernyi időtartam alatt tevékenységváltás történt. Mivel nincs igazán jó módszer egy (konvolúciós) neurális hálózat rétegeinek és paramétereinek meghatározására, így leginkább tapasztalatokra, illetve kísérletezésre támaszkodhatunk a struktúra kialakítása során. Az általam alkalmazott hálózat Matlab segítségével került implementálásra, amely az aktivációs függvényt külön rétegként kezeli. A hálózat felépítése a 6. táblázatban került összefoglalásra.

6. táblázat. Alkalmazott CNN struktúrája

Réteg sorszáma	**Réteg típusa**
1	Bemeneti réteg (8x64)
2	Konvolúciós réteg (8 darab 3x3-as kernel)
3	Batch normalizációs réteg (8 csatorna)
4	Aktivációs réteg (ReLU)
5	Maximum pooling réteg (2x2-es szűrő)
6	Konvolúciós réteg (16 darab 3x3x8-as kernel)
7	Batch normalizációs réteg (16 csatorna)
8	Aktivációs réteg (ReLU)
9	Maximum pooling réteg (2x2-es szűrő)
10	Konvolúciós réteg (32 darab 3x3x16-os kernel)
11	Batch normalizációs réteg (32 csatorna)
12	Aktivációs réteg (ReLU)
13	Teljesen összekapcsolt réteg
14	Aktivációs réteg (Softmax)
15	Osztályozó réteg

A tanítás során momentummal ellátott sztochasztikus gradiens optimalizálót használtam, 0,01-es tanulási rátával.

Ezzel a hálózattal az MLP-hez hasonló, 70,9%-os osztályozási pontosságot sikerült elérni a három osztály esetén, mindezt lényegesen rövidebb, körülbelül 30 óráig tartó, 30 epoch-ból álló tanítási folyamattal. Ezen felül további időmegtakarítást jelentett a viszonylag időigényes jellemzőkinyerés elmaradása. A hálózat által nyújtott osztályozási teljesítmény, kizárólag a tesztadatokon mérve, a 7. táblázatban megadott konfúziós mátrixon látható.

7. táblázat. Konfúziós mátrix (3 osztály)

Tényleges osztály				
	Pihenés	128599	20839	27318
	Kézfej	10207	25281	4050
	Lábfej	9381	3147	28596
		Pihenés	Kézfej	Lábfej
		Modell általi besorolás		

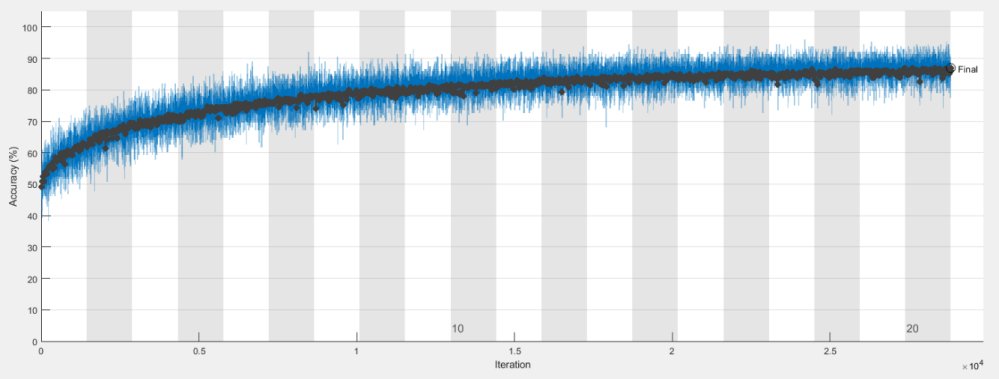
A konfúziós mátrixból az is kiolvasható, hogy az aktív tevékenységek (kézfej, illetve lábfej elképzelt mozgatása) esetén a hibás osztályozások között többszörös, két és fél, illetve háromszoros volt a pihenési feladattal történő összetévesztés mértéke a másik aktív tevékenységgel történő összetévesztéshez mérten, de eközben a pihenéshez tartozó adatminták száma körülbelül négy és félszeres volt. Ebből a megfigyelésből kiindulva

megvizsgáltam, hogy csak az aktív tevékenységek esetén, azaz a pihenést kizárva és így az osztályok számát kettőre redukálva milyen pontosság érhető el ugyanezzel a neurális hálózattal. Ebben az esetben, ahogy az a 8. táblázatban látható, 86,9%-os felismerési pontosságot kaptam.

8. táblázat. Konfúziós mátrix (2 osztály)

Tényleges osztály			
	Lábfej	34592	6532
	Kézfej	4035	35503
		Lábfej	Kézfej
		Modell általi besorolás	

A hálózat konvergenciáját a 27. ábra mutatja be. Látható, hogy a kék színnel jelölt, tanítási adatokon mért pontosság körülbelül azonos a tesztadatokon mért, fekete színnel jelzett pontossággal, így nem áll fenn a túltanulás (overfitting) jelensége, amelynek jellegzetessége, hogy a tesztadatokon lényegesen rosszabbul teljesít a hálózat, mint a tanítási adatokon.



27. ábra. Neurális hálózat konvergenciája PhysioNet adatokon

A hálózat tanítása során az alkalmazott epoch korlát 20 volt, és amint az az ábrán látható, a görbe alakja a tanítás végére ellaposodott, de úgy tűnik, hogy nem érte el a lehetséges maximumát. Úgy véltem, hogy a korlátot megnövelve valamelyest még javítható a hálózat teljesítménye. Végül az epoch korlátot 100-ra növelve a 9. táblázatban látható konfúziós mátrixhoz jutottam, amely 89,95%-os felismerési arányt jelent. Megjegyzendő azonban, hogy körülbelül a 70. epoch-tól már nem növekedett tovább a pontosság, így ez tekinthető a maximális értéknek, amelyet ezzel a hálózattal, ezeken az adatokon el tudtam érni.

9. táblázat. Konfúziós mátrix (2 osztály, 100 epoch)

Tényleges osztály			
	Lábfej	37912	3212
	Kézfej	4897	34641
		Lábfej	Kézfej
		Modell általi besorolás	

Az előbbieken túl kipróbáltam két további módosítást a bemenő adatokon, annak reményében, hogy a pontosság tovább növelhető valamilyen módon. Az egyik ilyen módosítás a bemenő adatmátrix sorainak felcserélését jelentette, úgy, hogy az jobban illeszkedjen az elektródák fizikai pozíciójához a fejbőrön. Az összes korábbi mérés során a 3. táblázatban ismertetett csatornasorrendendet használtam, azonban ezúttal ezek néhány különböző permutációját is kipróbáltam arra alapozva, hogy lényeges lehet a sorok egymáshoz viszonyított helyzete is a mátrixban lévő geometriai struktúrák szempontjából. Szemléletesen, például egy kép sorait felcserélve

megtörténhet, hogy nem lesz felismerhető rajta az eredeti objektum. Azt tapasztaltam azonban, hogy a csatornák sorrendjének cseréje nem változtatott a hálózat osztályozási teljesítményén.

A másik módosítás pedig eltérő csatornák használatát jelentette. Korábbi kutatások alapján tudjuk, hogy az elképzelt motoros tevékenységek az agyban a C3 és C4 elektródák alatti elsődleges szenzomotoros területeket aktiválják [62-63]. A perietális csatornák (pl. P3, P4, Pz) jelentőségéről szintén beszámoltak korábban [64]. Ezen kutatási eredmények felhasználásával összeállítottam egy új csatornalistát, amely a 10. táblázatban látható.

10. táblázat. Újonnan választott csatornák száma és elnevezése

Csatorna száma	**Csatorna neve**
1. csatorna	C3
2. csatorna	C4
3. csatorna	F3
4. csatorna	Fz
5. csatorna	F4
6. csatorna	P3
7. csatorna	Pz
8. csatorna	P4

Az újonnan kiválasztott csatornákból származó adatokkal újratanítottam a korábban alkalmazott hálózatot és így 91,3%-os felismerési pontosságot sikerült elérni a tesztadatokon. A tesztelés során kapott konfúziós mátrix a 11. táblázatban látható.

11. táblázat. Konfúziós mátrix új csatornákon (2 osztály, 100 epoch)

Tényleges osztály			
	Lábfej	38699	2425
	Kézfej	4618	34920
		Lábfej	Kézfej
		Modell általi besorolás	

Annak ellenére, hogy ez a csatornalista valamelyest jobb eredményeket biztosított, mint a korábbi, mégsem használtam ezt a jövőben, hanem maradtam az előző kombinációnál. Ennek oka az, hogy a későbbiekben szeretném a PhysioNet adatbázison betanított neurális hálózatot saját gyűjtésű mérési adatokon alkalmazni, és bár az általam használt EEG headset biztosít némi flexibilitást az elektródák fizikai elhelyezésére vonatkozóan, azonban közel sem akkorát, hogy ezt a jobbnak bizonyult kombinációt alkalmazni lehetne.

6.4. CNN hatékonyságvizsgálata saját adatokon

A saját mérési adatok használatának célja, hogy megvizsgáljam az olyan EEG-eszközök lehetséges alkalmazását, amelyek megfizethetőek, és így segíthetik például fogyatékkal élők vagy idősek életét. Ehhez a korábban bemutatott OpenBCI Ultracortex Mark IV EEG headset-et, Cyton digitalizáló kártyát és az általam fejlesztett mérésvezérlő és adatgyűjtő szoftvert használtam. A méréseket a PhysioNet adatbázis esetén alkalmazottal megegyező módon végeztem: négy másodpercig tartottak az aktív feladatok, majd négy másodpercig a pihenés. Ez 15 iteráción keresztül folytatódott, így

két perc hosszúságú méréseket kaptam eredményül. A mérés 10 alkalommal került megismétlésre, ezáltal összesen húsz percnyi EEG felvételhez jutottam, amelyek egy főtől származtak.

A mért adatok vizuális ellenőrzését és trendmentesítését követően itt is 8x64-es mátrixokra bontást végeztem, az előző fejezetben ismertetett módon, majd eltávolítottam azokat a mátrixokat, amelyekben tevékenységváltás történt. Mivel a rendelkezésre álló adatmennyiség a PhysioNet adatbázishoz viszonyítva elég csekély volt, ezért a mátrixok között átfedést alkalmaztam, minden mátrix első 63 oszlopa azonos volt a megelőző mátrix utolsó 63 oszlopával. Természetesen ez a módszer nem ideális, hiszen az egymást követő minták kevés új információt tartalmaznak egymáshoz képest, így elsősorban a tanulási idő növekszik, azonban valamelyest mégis képes növelni a rendelkezésre álló információmennyiséget.

Némi különbség adódik abból eredően, hogy a PhysioNet adatbázis esetén 160 Hz a mintavételi frekvencia, így a 64 minta 0,4 másodpercet fog át, míg az OpenBCI eszköznél ez 250 Hz, azaz körülbelül negyed másodpercnyi időtartamot fed le a bemenő adatmátrix. Mivel a két mintavételi frekvencia hányadosa nem egész szám, ezért az azonos frekvenciára hozás egy kiküszöbölendő probléma, ha a PhysioNet adatbázison tanított hálózatot szeretnénk saját adatok felismerésére használni. Ennek egy lehetséges megoldása a jel újramintavételezése az új frekvenciával, majd egy véges impulzusválaszú szűrő alkalmazása és a szűrő által okozott késés kompenzációja.

Az alkalmazott CNN struktúrája és a tanítási paraméterek megegyeztek a PhysioNet adatbázis esetén használttal. A tanítás kiegyensúlyozott adatkészleten történt, amelynek 70%-át használtam fel tanítási mintaként, a többi pedig tesztelési célokat szolgált, ideértve a kiegyensúlyozás során a

tanítási halmazból kikerült adatokat is. A tanítás 20 epoch-ig futott, amelyet valós időben figyelve, a hálózat konvergenciája alapján elegendőnek ítéltem. Az általam használt Intel Core i7-4790 processzort, 8 GB RAM-ot és NVIDIA GeForce GTX 860M GPU-t tartalmazó számítógépen ez körülbelül 41 órát vett igénybe, annak ellenére, hogy párhuzamos végrehajtási módot alkalmaztam GPU-n futtatva. A tesztadatokra vonatkozó konfúziós mátrixot a 12. táblázat mutatja be.

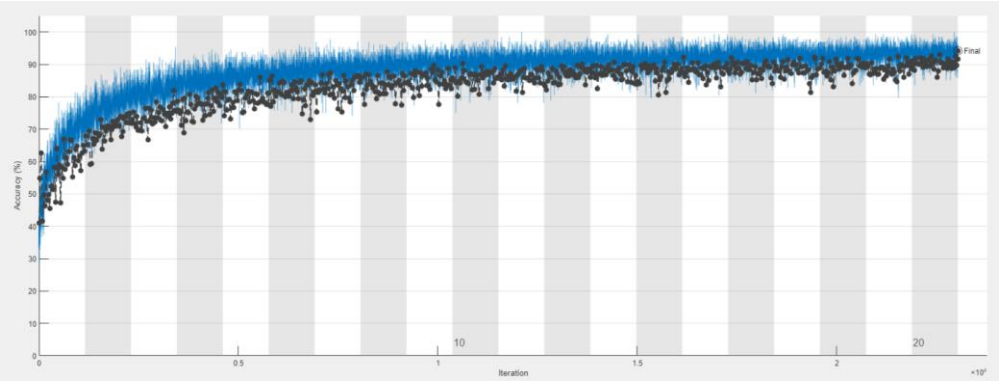
12. táblázat. Konfúziós mátrix (3 osztály)

Tényleges osztály				
	Kézfej	20539	41	502
	Lábfej	134	18879	2069
	Pihenés	3263	1694	86400
		Kézfej	Lábfej	Pihenés
		Modell általi besorolás		

Az adatokból látható, hogy a neurális hálózat tévesztésének mértéke az aktív tevékenységek esetén kissé más képet mutat, mint a PhysioNet adatbázis használatánál. A kézfej mozgatásakor bekövetkező hibás osztályozások között több, mint tizenkétszeres a különbség a pihenés javára a lábfejjel szemben, míg a lábfej mozgatásakor ez az arány már tizenötszörös a pihenés és a kézfej mozgatása között. Eközben a pihenéshez tartozó adatmennyiség csupán négyszeres, így ez azt jelenti, hogy a PhysioNet adatok használatánál a

pihenést jobban el lehetett különíteni az aktív tevékenységektől, mint saját mérési adataim esetén.

A mátrix adataiból az is látható, hogy a hálózat által elért osztályozási pontosság rendkívül nagy, 94,2%. Tevékenységekre lebontva ez 97,4% a kézfej, 89,6% a lábfej és 94,6% a pihenés esetén. Megjegyzendő, hogy valamelyest nagyobb mértékű túltanulás jelentkezett, mint a PhysioNet adatok esetén, ahogy az a 28. ábrán is látható, de az így sem volt jelentős, a tesztadatokon mért pontosság elég jól megközelítette a tanítási adatokon adódó pontosságot.



28. ábra. Neurális hálózat konvergenciája saját mérési adatokon

Bár a kapott osztályozási pontosság azt mutatja, hogy egy ilyen, nem klinikai, hanem fogyasztói szegmensbe szánt, olcsóbb EEG eszközzel is jó eredményeket lehet elérni megfelelő gépi tanulási láncot alkalmazva, azonban azt az eredmények értékelésénél mindenképp ki kell emelni, hogy az adatok egy főtől származtak, ami közel sem ideális. Amint azt korábban megmutatták, a vizuális és motoros kéreg agyi tevékenységei erősen egyénfüggőek [65]. Ebből következően több önkéntestől származó adatok felhasználásával a kapott eredmények valamelyest rosszabbak lettek volna, hiszen a tanulóalgoritmusnak egy általánosabb jellegű mintázatot kell megtalálni az adatokban. Ezzel együtt is kijelenthető azonban, hogy ilyen jellegű feladatokra

az olcsóbb EEG eszközök is kiválóan felhasználhatók, amely például mozgásképtelen személyek életvitelének segítésére is alkalmas lehet a jövőben.

A PhysioNet és a saját gyűjtésű adatbázis adataival elért eredmények ismeretében érdemes összefoglalni azokat a tényezőket, amelyek szerepet játszhatnak abban, hogy eltérő pontossággal lehetett elkülöníteni a különböző tevékenységeket a két adatforrás használata esetén. A tapasztalt különbségnek több oka is lehet, bár ezek olykor ellentétes hatásúak. A PhysioNet adatbázis esetén egy professzionális, laboratóriumi eszközzel történt az adatfelvétel, ez alapján valószínűleg jobb minőségűek az adatok, amely nagyobb pontosságot indukál. A mintavételi frekvencia is eltérő volt a PhysioNet és a saját mérések esetén, így az alkalmazott, 64 mérési pontból álló ablakméret PhysioNet esetén nagyobb időtartamot fog át, amely a korábban tárgyalt, ideális szegmensméretre vonatkozó megállapítások alapján szintén nagyobb pontosságot feltételezne. Ezzel szemben, ami a saját adatokon elérhető jobb eredményeket segíti, az az alanyok mérés közbeni magatartása és a mérés körülményei. A saját gyűjtésű adatok esetén magamon végeztem el az adatfelvételt, odafigyelve arra, hogy az objektumok megjelenésekor azonnal, és jól reagáljak. Tévesztés esetén a keletkezett mérési fájlt megsemmisítettem, továbbá a mérések között, ha szükségét éreztem, pihentem. A mérőeszközt eközben nem távolítottam el, így az elektródák helyzete nem változott meg a mérés során. Ezzel szemben a PhysioNet adatbázis esetén több különféle mérést végeztek, így az hosszabb ideig tartott, nehezebb volt végig fenntartani a teljes koncentrációt, növekedett a reakcióidő. Mivel összesen 109 alanyon végezték el az adatgyűjtést, megtörténhet, hogy nem volt arra idő, hogy szüneteket tartsanak, amikor elkezdett fáradni az önkéntes, továbbá kérdéses az is, hogy ha egyszer-kétszer tévesztettek a mért személyek, azt jelezték-e a mérés után az adatok újrafelvétele érdekében. Emellett ugyanakkor van egy

másik, jelentősebb különbség a saját adatokon végzett mérés javára, amely a kísérletben részt vevő alanyok számából és az adatok egyénfüggőségéből fakad. Valószínűleg az itt alkalmazott neurális hálózati architektúra nem teljesít túl jól az általános jellegű, egyénfüggetlen minták megtalálásában – ahogy az a dolgozat ezt követő alfejezetében látható lesz, sikerült végül olyan architektúrát is találnom, amely azonos, 64 mintás ablakméret esetén lényegesen jobb eredményt biztosít. Mindezek a tényezők együttesen megmagyarázzák, hogy miért jelentkezett különbség az elért pontosságban a saját és a PhysioNet adatbázison végzett mintafelismerések között az azonos gépi tanulási modell használata ellenére. Az itt ismertetett eredmények alapján került megfogalmazásra a negyedik tézispont.

4. tézis: *Megmutattam, hogy egy alacsony költségű, nem klinikai alkalmazásra szánt EEG headset által biztosított adatok minősége – megfelelő előfeldolgozást követően – jól használható tevékenységfelismerésre. Az általam alkalmazott CNN architektúrával 94,2%-os osztályozási pontosságot sikerült elérni három osztályra vonatkoztatva.*

6.5. CNN struktúrák összehasonlítása PhysioNet adatokon

Időközben sikerült beszerezni egy új, OpenBCI EEG Electrode Cap Kit EEG eszközt és a Cyton mellé egy Daisy kiegészítő kártyát, amellyel így 16 csatornán lehet méréseket végezni. A rendelkezésre álló új csatornalistát a 13. táblázat foglalja össze. A továbbiakban ezen csatornák felhasználásával végeztem különböző struktúrájú CNN-ek hatékonyságvizsgálatát.

13. táblázat. Csatornák száma és elnevezése a 16 csatornás eszköz esetén

Csatorna száma	Csatorna neve
1. csatorna	Fp1
2. csatorna	Fp2
3. csatorna	F7
4. csatorna	Fz
5. csatorna	F8
6. csatorna	T7
7. csatorna	C3
8. csatorna	Cz
9. csatorna	C4
10. csatorna	T8
11. csatorna	P7
12. csatorna	P3
13. csatorna	P4
14. csatorna	P8
15. csatorna	O1
16. csatorna	O2

A neurális hálózat struktúrájának változtatásán túl kipróbáltam különböző szegmensméretek hatását is az osztályozási pontosságra. Ezt korábban már megtettem MLP esetén is a 4.1. fejezetben leírtak szerint, azonban CNN-nél még nem került rá sor. Az alkalmazott neurális hálózatok közül az első (a továbbiakban *CNN1*) egy tisztán konvolúciós hálózat, azaz nem tartalmaz teljesen összekapcsolt rétegeket. Felépítése a 14. táblázatban került összegzésre.

14. táblázat. CNN1 hálózat struktúrája

Réteg sorszáma	**Réteg típusa**
1	Bemeneti réteg (szegmensméret-függően)
2	Konvolúciós réteg (16 darab 5x5-ös kernel, 2-es lépésköz, párnázással)
3	Batch normalizációs réteg (16 csatorna)
4	Aktivációs réteg (ReLU)
5	Konvolúciós réteg (32 darab 5x5-ös kernel, 2-es lépésköz, párnázással)
6	Batch normalizációs réteg (32 csatorna)
7	Aktivációs réteg (ReLU)
8	Konvolúciós réteg (64 darab 3x3-as kernel, 2-es lépésköz, párnázással)
9	Batch normalizációs réteg (64 csatorna)
10	Aktivációs réteg (ReLU)
11	Konvolúciós réteg (64 darab 2x8-as kernel, 2x8-es lépésköz, párnázás nélkül)
12	Batch normalizációs réteg (64 csatorna)
13	Sorosító réteg

A tanítás – és a tesztelés is – kiegyensúlyozott adathalmazon történt, Adam optimalizálóval, kezdetben 10 fő adatainak felhasználásával. Első alkalommal 32 mintás, azaz 0,2 másodperces ablakméretet alkalmaztam, amellyel 79,2%-os felismerési pontosságot sikerült elérni. Az egyes osztályokra vonatkozó értékeket az A. függelék 1. táblázata tartalmazza. Annak ellenére, hogy ez egy más struktúrájú hálózat, itt is megfigyelhető az a

korábbi tapasztalatom, hogy az aktív tevékenységeket jóval kisebb arányban téveszti össze egymással a hálózat, mint a pihenéssel.

Elvégeztem ugyanezt a kísérletet 20 fő adatain is, és a kapott eredmények megerősítették korábbi feltevésemet, illetve a [65] cikkben leírtakat, miszerint az ilyen jellegű agyi aktivitások egyénenként változóak. Az összesített eredmény 71,8%-ra adódott, amely jelentősen elmarad a hálózat 10 fő adataival nyújtott teljesítményétől. A hálózat konfúziós mátrixát az A. függelék 2. táblázata mutatja meg.

A hálózat hatékonyságát ilyen kis ablakméret mellett nem találtam kielégítőnek, de 64-es méretű, azaz 0,4 másodperces szegmensekkel ez nagymértékben javult, 10 főnél 91,1%-ra, 20 főnél 83,3%-ra növekedett a pontosság, ahogy azt részletesen az A. függelék 3. és 4. táblázata mutatja.

A szegmensek méretét tovább növelve 128 mintára (0,8 másodperc), még jobb osztályozási eredményeket értem el, 96,8% (10 fő) és 94,6%-ot (20 fő). Azt tapasztaltam, hogy ekkora ablakméretnél már nincs jelentős különbség a két adathalmazon elérhető eredmény között. A konfúziós mátrixok az A. függelék 5. és 6. táblázatában találhatók.

Az utolsó alkalmazott ablakméret 160 volt, amely 1 másodpercet fed le, és valós idejű adatfeldolgozásnál is ekkora késleltetést okoz, így ezt már nem kívántam tovább növelni. Az osztályozási pontosság 10 fő esetén 99,1%-ra, míg 20 főnél 97,7%-ra adódott. Az eredményeket úgy értelmeztem, hogy ilyen méretű szegmens már elegendő egyénfüggetlen információt tartalmaz ahhoz, hogy egy általános mintázat felismerésére biztosítson lehetőséget a gépi tanulási modell számára. Az eredmények részletesen az A. függelék 7. és 8. táblázatában kerültek összegzésre.

Elvégeztem ugyanezeket a kísérleteket egy más felépítésű, összevonó és teljesen összekapcsolt rétegeket is tartalmazó hálózat (a továbbiakban *CNN2*) használatával, melynek felépítése a 15. táblázatban került bemutatásra.

15. táblázat. CNN2 hálózat struktúrája

Réteg sorszáma	**Réteg típusa**
1	Bemeneti réteg (szegmensméret-függően)
2	Konvolúciós réteg (8 darab 3x3-as kernel, 1-es lépésköz, párnázással)
3	Aktivációs réteg (ReLU)
4	Maximum pooling réteg (2x2-es szűrő, 2-es lépésköz)
5	Konvolúciós réteg (16 darab 5x5-ös kernel, 1-es lépésköz, párnázás nélkül)
6	Aktivációs réteg (ReLU)
7	Maximum pooling réteg (2x2-es szűrő, 2-es lépésköz)
8	Sorosító réteg
9	Teljesen összekapcsolt réteg (64 neuron)
10	Aktivációs réteg (ReLU)
11	Teljesen összekapcsolt réteg (32 neuron)
12	Aktivációs réteg (ReLU)
13	Teljesen összekapcsolt réteg (3 neuron)
14	Aktivációs réteg (Softmax)

Azt tapasztaltam, hogy ez a hálózat a *CNN1*-hez képest lényegesen rosszabbul teljesít. 32-es ablakméretnél 10 fővel 62,5%, 20 fővel 58,4%-os pontosságot sikerült elérni – ez utóbbihoz tartozó konfúziós mátrix az A.

függelék 9. táblázatában található. A továbbiakban a 10 fős eredményekhez tartozó részletes mátrixokat – terjedelmi okokból – nem közlöm, kizárólag az összesített eredményt.

Nagyobb, 64-es ablakmérettel a pontosság valamelyest növekedett, azonban az így kapott érték még mindig lényegesen elmarad a *CNN1* teljesítményétől. A kapott konfúziós mátrix az A. függelék 10. táblázatában látható. A szegmensméretet 128-ra növelve továbbra is mintegy 20 százalékponttal alacsonyabb pontosság érhető el a *CNN1*-hez viszonyítva, és ez a tendencia 160-as ablaknál is megmarad. A 20 fővel kapott konfúziós mátrixok ez utóbbi két szegmensméret esetén az A. függelék 11. és 12. táblázatában kerültek bemutatásra. Az eredményekből látható, hogy az ilyen típusú adatokban rejlő jellemzőket jobban megtalálja a *CNN1* architektúra, így ehhez az adatkészlethez jobban illik, a *CNN2* kevésbé eredményesen tudja ezt megtenni. Ebben szerepet játszik az is, hogy a *CNN1* több konvolúciós rétege jobban finomhangolható a több paraméternek köszönhetően, így egy jó paraméterkombináció a hatékonyságot jelentősen javítani tudja, míg a teljesen összekapcsolt rétegek esetén ez a lehetőség korlátozottabb.

Mivel a *CNN1* és *CNN2* közül az első lényegesen jobb osztályozási pontosságot biztosított, ezért alapvetően ahhoz a struktúrához visszatérve egy másik megközelítést alkalmazva végeztem el az újabb kísérleteket. Ebben a hálózatban (a továbbiakban *CNN3*) az összes konvolúciós rétegben a szűrők méretét 5x5-ösről 3x3-as mérsékeltem, minden más tekintetben a modell azonos a *CNN1*-gyel.

A *CNN3* hálózat esetén a legkisebb szegmensméretet alkalmazva a *CNN1* által elért értékeket megközelítő, azonban némileg, körülbelül 3 százalékponttal rosszabb teljesítményt kaptam. Nagyobb ablakméretek használatánál is hasonló a két hálózat teljesítményének viszonya, a *CNN1* kissé

jobb pontosságot nyújt, mint a *CNN3*. A 20 fős mintákkal a négy különböző ablakméret esetén kapott konfúziós mátrixok az A. függelék 13-16. táblázataiban találhatók. Az eredményekből látható, hogy konzekvensen valamelyest rosszabbul teljesített ez az architektúra, mint a *CNN1*, így arra következtethetünk, hogy ezekkel az adatokkal, ezt a hálózati rétegrendet alkalmazva az 5x5-ös kernelméret használata kedvezőbb, mint a 3x3-as.

A tapasztalatok alapján a következő alkalmazott modellel (a továbbiakban *CNN4*) ismét a *CNN1*-ben használt kernelmérethez tértem vissza, ezúttal azonban a hálózat mélyítésének hatását vizsgáltam meg. A *CNN1* hálózatot alapvetően egy konvolúciós – batch normalizációs – ReLU rétegeket tartalmazó blokkal egészítettem ki, amint az a 16. táblázatban látható.

16. táblázat. CNN4 hálózat struktúrája

Réteg sorszáma	**Réteg típusa**
1	Bemeneti réteg (szegmensméret-függően)
2	Konvolúciós réteg (8 darab 5x5-ös kernel, 2-es lépésköz, párnázással)
3	Batch normalizációs réteg (8 csatorna)
4	Aktivációs réteg (ReLU)
5	Konvolúciós réteg (16 darab 5x5-ös kernel, 2-es lépésköz, párnázással)
6	Batch normalizációs réteg (16 csatorna)
7	Aktivációs réteg (ReLU)
8	Konvolúciós réteg (32 darab 5x5-ös kernel, 2-es lépésköz, párnázással)
9	Batch normalizációs réteg (32 csatorna)
10	Aktivációs réteg (ReLU)

11	Konvolúciós réteg (64 darab 3x3-as kernel, 2-es lépésköz, párnázással)
12	Batch normalizációs réteg (64 csatorna)
13	Aktivációs réteg (ReLU)
14	Konvolúciós réteg (64 darab 2x8-as kernel, 2x8-es lépésköz, párnázás nélkül)
15	Batch normalizációs réteg (64 csatorna)
16	Sorosító réteg

Ezzel a modellel is azonos kísérleteket végeztem el, mint az előbbiekkel. A 20 fős adatbázison a négy különböző szegmensmérettel kapott konfúziós mátrixokat az A. függelék 17-20. táblázatai tartalmazzák. Az adatokból látható, hogy ez a mélyebb hálózat nem biztosított jobb eredményeket a sekélyebb változathoz képest, teljesítménye kismértékben elmaradt tőle.

Az említett hálózatok és ablakméretek alkalmazásával kapott eredményeket a 17. táblázat összegzi a 10 és 20 fős adatkészletek esetén.

17. táblázat. PhysioNet adatokon elért pontosság 10 és 20 fő esetén

Hálózat	**Ablakméret (mintaszám és időtartam)**			
	32 (0,2 mp)	64 (0,4 mp)	128 (0,8 mp)	160 (1 mp)
CNN1	79,2% / 71,8%	91,1% / 83,3%	96,8% / 94,6%	99,1% / 97,7%
CNN2	62,5% / 58,4%	62,1% / 64%	76,4% / 74,4%	82,6% / 76,4%
CNN3	76,5% / 68,6%	87,8% / 80,2%	96,4% / 91,2%	97,7% / 93,6%
CNN4	76,2% / 70,4%	86,1% / 79,8%	96,9% / 92,9%	99% / 96,1%

A táblázat adataiból látható, hogy erős pozitív korreláció van a szegmensméret és a hálózat osztályozási hatékonysága között, azaz a szegmensméret növelésével jelentősen nőtt a gépi tanulási modell teljesítménye, bármelyik alkalmazott hálózatot is tekintjük. Megállapítható továbbá, hogy a batch normalizációt nem, viszont összevonó és teljesen összekapcsolt rétegeket tartalmazó *CNN2* teljesítménye mindig lényegesen alulmaradt a többi hálózatéhoz viszonyítva.

Az kernelméret változtatásának hatását tekintve az 5x5-ös méretet használó *CNN1* minden esetben valamelyest jobban teljesített, mint a 3x3-ast alkalmazó *CNN3*. A [66] cikkben leírtak alapján ugyan az ideális kernelméret személyenként változik, sőt, akár egy adott egyén esetén is más és más lehet időről időre, azonban megállapítható, hogy ezen az adatkészleten, ezzel a neurális hálózati struktúrával egy több személyre általánosítható megoldást keresve, jobb választásnak bizonyul a nagyobb méretű szűrő alkalmazása.

A hálózat mélységének tekintetében elmondható, hogy egy mélyebb hálózat alkalmazása nem feltétlenül előnyösebb egy sekélyebbhez képest. Egy több konvolúciót tartalmazó hálózat elméletileg több releváns jellemzőt nyerhet ki, ilyen módon jobb osztályozási teljesítményt nyújthat. Ellene szól ugyanakkor, hogy a több paraméter miatt több időt igényel a tanítása, illetve hajlamosabb a túltanulásra, amelynek a végeredménye egy kevésbé általánosítható modell. A táblázat adataiból látható, hogy a sekélyebb *CNN1* jobban teljesített ebben a feladatban, mint a mélyebb *CNN4*.

A konfúziós mátrixok segítségével megállapítható az is, hogy általánosságban a modellek jobban el tudták különíteni az aktív feladatokat (kézfej, illetve lábfej elképzelt mozgatása) egymástól, mint a pihenéstől, bár itt – különösen a kisebb szegmensméretek esetén – néhány ellenpéldát is találunk. Az előbbi eredmények alapján fogalmaztam meg az ötödik tézispontot.

5. tézis: *A PhysioNet EEG adatbázis adatainak felhasználásával, méréseim során megmutattam, hogy optimális szegmensméret és architektúra megválasztással a neurális háló felismerési aránya egyértelműen növelhető. A legjobb eredmények a tesztadatokon mérve 99,1% és 97,7%-ra adódtak 10, illetve 20 fő adatainak felhasználásával három tevékenységet magában foglaló osztályozás esetén. Megállapítottam továbbá, hogy egy hálózat indokolatlan mélyítése nemcsak, hogy nem növeli tovább, hanem akár csökkentheti is az osztályozási pontosságot – amellett, hogy a tanítási időigénye is nagyobb.*

7. Hardveres implementáció

Kutatásom lezárásaként az általam alkalmazott MLP és CNN neurális hálózatok hardveres gyorsítási lehetőségeit vizsgáltam meg FPGA felhasználásával. Ehhez kétféle megközelítést alkalmaztam, melyek közül az egyik egy teljes mértékben saját fejlesztésű, hardverleíró nyelven készített MLP implementáció, míg a másik egy Xilinx Alveo U50 gyorsítókártyán implementálható DPU (*deep learning processing unit*) felhasználásával történt.

7.1. MLP hálózat FPGA implementációja

Egy betanított MLP hálózat definiálásához a hálózat struktúráján túl a neuronoknál alkalmazott súlyok és eltolási értékek ismerete szükséges. A legtöbb gépi tanulási keretrendszer, ideértve az általam is alkalmazott ROOT, Matlab és Keras környezetet, képes arra, hogy ezeket valamilyen formában exportálja. MLP hálózat FPGA-n történő implementációja céljából készítettem egy Python programot, amely ezen értékek felhasználásával képes szintetizálható Verilog nyelvű hardveres leírást generálni a neurális hálózatról. Az általam készített program egy egyszerű szöveges fájlból tudja kigyűjteni a súlyokat és eltolási értékeket, amely bármely előbb említett környezetben elkészíthető.

Míg szoftveres megvalósítás esetén a lebegőpontos számábrázolás használata semmilyen problémát nem okoz a processzor lebegőpontos egységének köszönhetően, addig egy FPGA-s implementáció esetén ilyen számokkal műveleteket végezni igen nehézkes. Természetesen megoldható lebegőpontos számítási egységek implementációja is, azonban ettől sokkal célravezetőbb, ha fixpontos számábrázolást alkalmazunk. Az általam készített

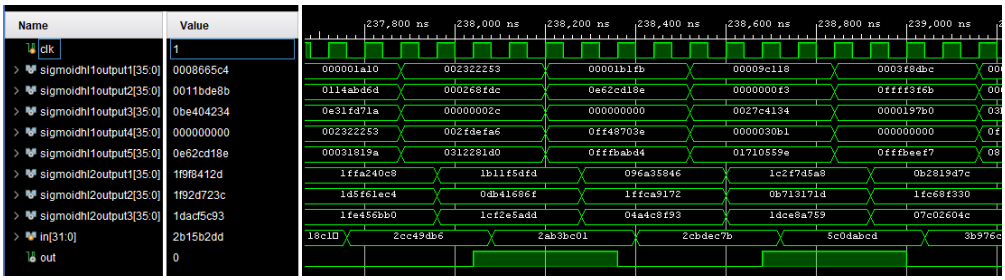
Python program a súlyokat és eltolási értékeket fixpontossá alakítja, majd tárolásukhoz memóriát generál. A neuronok implementációjánál szintén adott bitszélességű fixpontos adatokkal dolgozó neuronokat hoz létre, ez a bitszélesség paraméterként megadható. Az aktivációs függvényt LUT-ként (*lookup table*) implementálja memóriában, ahol a memóriacím a függvény független változójának értéke, és a cím által kijelölt cella tartalmazza a függvényértéket. A LUT méretének kordában tartása érdekében megoldható (paraméterként megadható), hogy a neuron által számított súlyozott összeg bitjeinek csak egy részét, például a legnagyobb helyiértékű 12 bitet használjuk fel a memória címzésére. A LUT tároláshoz dual-port ROM-ot használtam, így két neuron olvashatja egyidőben ugyanazt a memóriát, felére csökkentve ezzel ezt a fajta az erőforrásigényt.

Az implementáció optimalizálható nagyobb sebességre vagy kisebb erőforráshasználatra. Az általam készített program olyan Verilog kódot generál, amely a nagyobb sebességet helyezi előtérbe. Ez a gyakorlatban azt jelenti, hogy minden neuron minden bemeneténél egy szorzó található, amelyek egymással párhuzamosan működnek, majd a szorzatok összegzése is egy lépésben történik. Ilyen megközelítés alkalmazásánál egy két rejtett réteget, rétegenként öt, illetve három neuront tartalmazó, 32 egybites bemenettel rendelkező hálózat 184 digitális jelfeldolgozó (*digital signal processing, DSP*) blokkot használt fel, amely bizonyos FPGA-k esetén soknak bizonyulhat. Alternatívaként megvalósítható olyan implementáció, amely neurononként egy DSP egységet használ, a bemeneti értékek szorzását, majd akkumulálását egyesével, szekvenciálisan végezve. Ez egy lassabban működő, de lényegesen kisebb erőforrásigényű megoldás. Ezen kívül köztes megoldásként lehetséges ezen megközelítések kombinációja is, amely egy részlegesen párhuzamos működést valósít meg.

A neurális hálózatok hardveres implementációja egy kurrens téma, így a kutatók részéről több kezdeményezés is történt MLP hálózat FPGA-alapú létrehozására, azonban az általam fellelt implementációk egyike sem pontosan ilyen, így egyedinek tekinthető. Például a [67] publikációban szintén MLP-t implementáltak FPGA-n, de ez manuálisan történt, Xilinx System Generator használatával, így különböző felépítésű hálózatoknál egyedileg kell módosításokat végezni, ezzel szemben az én megoldásom automatikusan adaptálódik a kívánt struktúrához. Emellett én a szigmoid függvény implementációjára LUT-ot, míg ezen cikk szerzői egy közelítő számítási módszert alkalmaztak. A [68] publikáció szerzői VHDL nyelven készítettek MLP hálózatot, amelyben saját implementációmhoz hasonlóan, LUT-ként történt a szigmoid függvény megvalósítása, azonban a neuronok felépítése eltért. A bemeneti értékek súlyokkal történő szorzása egyesével szekvenciálisan történt, míg az én implementációm párhuzamosan végzi, emellett hiányzik belőle az automatizált kódgenerálás lehetősége is. Külön kiemelném, hogy az én megoldásom a szoftveres betanított neurális hálózathoz a struktúra megadását követően automatikusan mindent elvégez és legenerálja a Verilog kódot. A felhasználó dolga mindössze annyi, hogy egy fejlesztői környezetben létrehozott projekthez ezeket a fájlokat hozzáadja és elindítsa az FPGA konfigurációs fájl generálását – bár a legtöbb fejlesztői környezetben ezen lépések is automatizálhatók egy TCL kód létrehozásával.

A hardveres implementáció megfelelő működését a szoftveres úton számolt értékekkel történő összehasonlítással bizonyítottam. 2000 véletlenszerűen generált tesztesetre kiszámoltattam a szoftveres hálózat kimenetét, ezeket egy fájlba mentettem, majd készítettem egy programot, amely a fájl tartalma alapján legenerálta a Verilog tesztkódot. A tesztesetek közül mindössze háromszor adott eltérő osztályozási eredményt a hardveres és a szoftveres implementáció. Ezeket alaposabban megvizsgálva azt

tapasztaltam, hogy küszöbértékhez nagyon közel volt a kimeneti érték, és egyszerűen a kisebb számábrázolási pontosságból fakadt, hogy a hardveres implementációjú hálózat másképp osztályozott egy bemeneti mintát, mint a szoftveres. Az elvégzett szimulációról egy képernyőképet a 29. ábra prezentál, amelyen megmutatja a hálózat bemenetét, kimenetét, illetve néhány belső jel értékét (szigmoid aktivációs függvények kimenete). Az ábrán látható az is, hogy a hálózat teljes, be- és kimenet közti késleltetése négy órajel.



29. ábra. Hardveres MLP működését bemutató szimuláció

Az MLP hálózat hardveres implementációja adja az értekezés hatodik tézispontját.

6. tézis: *Kifejlesztettem egy szoftvert, amely képes szintetizálható Verilog kódot generálni MLP hálózat FPGA-s implementációjához. A szoftver a neurális hálózat tanítását végző keretrendszerből exportált súlyok és eltolási értékek alapján – az architektúra definiálását követően – elvégzi a lebegőpontos-fixpontos konverziókat, legenerálja a szorzókat, összeadókat, aktivációs függvényeket és ezek összekapcsolódását megvalósító Verilog kódokat, melyek tetszőleges FPGA fejlesztői környezetben importálhatók és szintetizálhatók. Mindez teljesen automatizált módon történik, így kapcsolódó szakértelem nélkül is egyszerűen elkészíthető egy hardveres implementáció. A módszer helyességét szimulációs eljárással igazoltam.*

7.2. CNN hálózat DPU implementációja

A Xilinx University Program keretén belül sikerült hozzájutni egy Alveo U50 gyorsítókártyához, amely – többek között – neurális hálózatok mintafelismerési sebességének gyorsításához használható. A kártya egy egyedi UltraScale+ FPGA-t tartalmaz, amely kizárólag Alveo architektúrán működik [69]. Alveo kártyára történő fejlesztéshez a gyártó a Vitis AI környezetet biztosítja, amelynek segítségével hardveresen gyorsítható a gépi tanulási modell működése.

Ahhoz, hogy a már betanított neurális hálózat működtethető legyen az Alveo kártyán, néhány folyamatlépés elvégzése szükséges, amelyek a 30. ábrán kerültek összegzésre.

30. ábra. DPU implementáció lépési

Ezen lépések közül az első a tanított modell „fagyasztott” gráfjának létrehozása. A létrehozott modell a legtöbb keretrendszer esetén tartalmaz olyan információkat (pl. a gradiensek értéke), amelyek lehetővé teszik a modell újbóli betöltését és például a tanítását onnan folytatva, ahol az abba lett hagyva, azonban ezek nem szükségesek a felismerés során. Ez a fagyasztási lépés eltávolítja az ilyen jellegű információkat, viszont megtartja azokat, amelyek fontosak, például maga a gráf felépítése, súlyok, stb., és egy speciális, Google Protocol buffer (.pb) formátumú fájlba menti el őket.

A DPU kizárólag fixpontos műveletek elvégzésére képes, ezért a következő lépés a fagyasztott gráf 32 bites lebegőpontos értékeinek 8 bites egészekre történő konvertálása. A fixpontos modell alacsonyabb memória sávszélességet igényel, továbbá nagyobb sebességet és energiahatékonyságot

biztosít, mint a lebegőpontos. A kvantálási kalibrálási folyamat címkézetlen bemenő adatokat (néhány ezer mintát) igényel, amelyek segítségével elemzi a kvantáló az értékek eloszlását, ilyen módon dinamikusan alkalmazkodni tud hozzá. A kvantálás során értelemszerűen a pontosság mindenképpen csökken némileg, azonban a kalibrálási folyamat lehetővé teszi, hogy ez ne legyen túl nagy mértékű. A kvantálás által a felismerési pontosságra gyakorolt negatív hatás a 18. táblázatban került bemutatásra. A vizsgálat a 20 fős adatbázison történt, 160 mintás ablakméret felhasználásával. Látható, hogy a módszer a felsorolt esetekben átlagosan 2,7 százalékpontos csökkenést okozott a felismerési pontosságban a lebegőpontos számok alkalmazásával elért eredményekhez képest.

18. táblázat. Kvantálás hatása a PhysioNet adatokon elért pontosságra

Hálózat	**Felismerési pontosság kvantálás nélkül**	**Felismerési pontosság kvantálással**
CNN1	97,7%	94,7%
CNN2	76,4%	73,8%
CNN3	93,6%	90,4%
CNN4	96,1%	94,2%

A kvantált modell rendelkezésre állását követően el kell végezni annak az alkalmazott DPU utasításkészletére történő lefordítását, amely a Vitis AI fordító keretrendszerével történik. A modell topológiájának elemzése után a fordító egy belső számítási gráfot készít köztes reprezentációként és különféle optimalizációkat hajt végre, majd a DPU mikroarchitektúra alapján előállítja a lefordított modellt.

Az alkalmazott DPUv3E egy konvolúciós neurális hálózatokra optimalizált programozható motor, amely végre tudja hajtani a Vitis AI speciális utasításkészletének utasításait és így lehetővé teszi számos konvolúciós hálózat hatékony megvalósítását. A DPU IP-ként (*intellectual property*) áll rendelkezésre, amely implementálható az Alveo kártya FPGA-jában. Az általa támogatott műveletek többek között magukban foglalják a konvolúciót, dekonvolúciót, összevonást (maximum és átlag esetén is), a ReLU függvény támogatását és a batch normalizációt is [70].

Az Alveo kártyán futtatva a mintafelismerést megvizsgáltam a sebességbeli előnyt a CPU-n történő futtatáshoz képest. Az alkalmazott processzor Intel Core i7-9700KF típusú volt, a rendszermemória 64 GB, amellyel átlagosan 5271,5, míg az Alveo kártyával 29339,9 mintafelismerést sikerült elvégezni másodpercenként. Ez jelentős különbség az Alveo javára, azonban figyelembe kell vennünk a kvantálás következtében fellépő pontosságcsökkenést is. Ezt mérlegelve jelen alkalmazásban a gyorsítókártya használata nem jár előnnyel, hiszen egy valós idejű felismerés esetén is a mérési adatok érkezési intenzitása jóval az alatt a szint alatt marad, amelyet még CPU-s feldolgozással kezelni lehet. Más jellegű alkalmazásokban azonban jelentős potenciállal bír ez a megközelítés. Egy, az általam alkalmazottaknál jóval komplexebb neurális hálózatnál már igen időigényes lehet a mintafelismerés, ezáltal könnyen előfordulhat, hogy az adatok beérkezésének üteme már meghaladja a processzor által kezelhető maximumot, ilyen esetben a gyorsítókártya alkalmazása jó megoldást jelenthet még abban az esetben is, ha ez valamelyest a pontosság kárára válik.

8. Összefoglalás

Az értekezés elején áttekintést adtam az EEG-alapú tevékenységfelismerés kutatásának aktuális helyzetéről, a korábban elért eredményekről. Ez az áttekintés rávilágít arra, hogy a terület igen szerteágazó, az alkalmazható előfeldolgozási és gépi tanulási módszerek sokfélék, a közülük való választás közel sem egyértelmű, amelynek következtében számos kérdés és bizonytalanság merül fel egy ilyen jellegű feladat megoldásával kapcsolatban.

Kutatásom során részben saját gyűjtésű, részben nyilvánosan elérhető adatbázisból származó adatokkal dolgoztam. A saját mérési adatok biztosításához kifejlesztettem egy szoftvert, amely képes OpenBCI EEG eszközökről adatgyűjtést végezni. A szoftver egy objektumot jelenít meg véletlenszerűen a képernyő különböző részein, miközben fogadja, rögzíti és felcímkézi az EEG eszközről érkező adatfolyamot. Az ilyen módon gyűjtött adatok valamelyest több előfeldolgozást igényeltek, mint a PhysioNet adatbázisból származóak, ugyanis a mért értékek folyamatosan növekvő tendenciát mutattak, amelyet polinomillesztéssel küszöböltem ki. Ezzel együtt is sikerült megmutatnom, hogy egy olcsó, nem klinikai felhasználásra szánt Ultracortex Mark IV EEG headset is kellően jó minőségű adatokat szolgáltat ahhoz, hogy egy megfelelő gépi tanulási lánc bemeneteként használva őket nagy osztályozási pontosságot lehessen elérni.

Ha az adatok már rendelkezésre állnak, azokon valamilyen jellegű előfeldolgozást kell végezni. Ezen lépések egyike az adatok szegmentálása. Munkám során meghatároztam azt az optimális szegmensméretet, amellyel a gépi tanulási modell teljesítménye maximalizálható EEG-alapú tevékenységfelismerés esetén. Mind MLP, mind pedig különböző CNN modellek esetén azt láttuk, hogy a szegmensméret növelésével a felismerési

pontosság is szigorúan monoton növekvő egy másodperces ablakméretig. Egy valós idejű alkalmazásnál ettől nagyobb méretű szegmens a növekvő késleltetésből fakadóan már egyébként is problémás lehet, azonban méréseim azt mutatták, hogy további negatív hatásként a pontosság kis mértékű csökkenése is bekövetkezik. Ezek tükrében az egy másodperces szegmensméretet tartom optimálisnak ebben az alkalmazásban.

A kinyert jellemzők osztályozási pontosságra gyakorolt hatását is vizsgáltam. Megmutattam, hogy EEG adatok használata esetén jelentős pontosságnövekedés érhető el, ha a nyers adatok mellett vagy helyett a hullámsávokban mért teljesítményt, mint kinyert jellemzőt használjuk fel MLP tanulóalgoritmus bemeneteként. Az EEG jel abszolút teljesítményét kiszámítva az alfa, béta, gamma, delta és théta hullámsávokban, és ezeket használva bemenő adatként 91,5%-os, míg a nyers adatok közvetlen felhasználása 56,2%-os felismerési pontosságot ért el ugyanaz a hálózat. Kísérletet tettem arra is, hogy a nyers adatokon független komponens analízist végezzek, és az így kapott értékeket, illetve egy másik megközelítésben az ezen értékek alapján számított abszolút teljesítményt használjam fel a neurális hálózat bemenő adataként. Méréseim azt mutatták, hogy önmagában az ICA alkalmazása rosszabb eredményt ad a nyers adatokon vett PSD-hez képest, az ICA és PSD együttes alkalmazása pedig hasonlót, mint a PSD önmagában. Ezek alapján a független komponens analízis elvégzését nem tartom szükségesnek.

A PhysioNet EEG adatbázis adatainak felhasználásával elvégeztem több különböző MLP és CNN architektúra teljesítményének összehasonlítását. Megállapítottam, hogy a CNN automatikus jellemzőkinyerési képessége jól használható EEG jelek esetén, a nyers adatokon alkalmazott CNN architektúrák összességében jobb eredményt biztosítottak, mint a számos kutatásban is vizsgált, bemenetként PSD-t használó MLP.

Szintén a PhysioNet EEG adatbázis adatain megmutattam, hogy az architektúra jó megválasztással a neurális háló felismerési aránya egyértelműen növelhető. A rendelkezésre álló 64 csatorna adataiból mindössze 16-ot felhasználva a legjobb felismerési arányok a tesztadatokon mérve 99,1% és 97,7%-ra adódtak 10, illetve 20 fő adatainak felhasználásával három tevékenységet magában foglaló osztályozás esetén. Megállapítottam továbbá, hogy egy hálózat indokolatlan mélyítése nemcsak, hogy nem növeli tovább, hanem akár csökkentheti is az osztályozási pontosságot – amellett, hogy a tanítási időigénye is nagyobb.

Végül pedig, kutatásom lezárásaként megvizsgáltam, hogy egy betanított neurális hálózat hardveres gyorsítására milyen lehetőséget kínálnak az FPGA-k. A megvalósítás egyrészt történhet tisztán hardveres módon, amelyhez kifejlesztettem egy olyan szoftvert, amely képes szintetizálható Verilog kódot generálni MLP hálózat automatikus FPGA-s implementációjának céljából. A szoftver a neurális hálózat tanítását végző keretrendszerből exportált súlyok és eltolási értékek alapján – az architektúra definiálását követően – elvégzi a lebegőpontos-fixpontos konverziókat, legenerálja a szorzókat, összeadókat és aktivációs függvényeket megvalósító kódokat. A generált Verilog kódok tetszőleges FPGA fejlesztői környezetben importálhatók és szintetizálhatók.

A hardveres gyorsítás megvalósításának egy másik lehetséges módja egy olyan megközelítés, amelynek során az FPGA-ban nem magát a konkrét hálózatot implementáljuk, hanem egy kifejezetten neurális hálózatok futtatásra optimalizált célprocesszort (DPU) hozunk létre, amely aztán képes a saját utasításkészletére lefordított gépi tanulási modell kódját végrehajtani. Ez utóbbi módszer összességében komplexebb, azonban IP-ként rendelkezésre áll maga a DPU, továbbá fordítóprogram is elérhető a legnépszerűbb keretrendszerekhez, így felhasználói szempontból jól alkalmazható.

9. Summary

At the beginning of the thesis, I gave an overview of the current state of EEG-based activity recognition research and the previously achieved results. This overview highlights that the field is very diverse, the applicable preprocessing and machine learning methods that can be applied are numerous and the choice among them is far from clear, which leads to many questions and uncertainties about how to approach such a task.

In my research, I worked partly with data I collected myself and partly with data from publicly available databases. To provide my own measurement data, I developed a software capable of collecting data from OpenBCI EEG devices. The software displays a target randomly on different parts of the screen while receiving, recording and labeling the data stream from the EEG device. The data collected in this way required somewhat more preprocessing than those from the PhysioNet dataset, as the measured values showed a steadily increasing trend, which was eliminated by polynomial fitting. Even so, I was able to show that a low-cost, consumer-grade Ultracortex Mark IV EEG headset for non-clinical use can provide data of sufficiently high quality to achieve high classification accuracy using them as inputs to a suitable machine learning chain.

When the data are already available, some kind of preprocessing is needed to be performed on them. One of these steps is data segmentation. In my work, I determined the optimal segment size to maximize the performance of the machine learning model for EEG-based activity recognition. For several MLP and CNN models, we have seen that as the segment size increases, the recognition accuracy also increases strictly monotonically up to a one-second window size. In a real-time application, a segment larger than this can be problematic anyway due to the increasing delay, however, my measurements

showed that as an additional negative effect, a small decrease in accuracy also occurs. In light of these, I consider a segment size of one second to be optimal for this application.

The impact of the extracted features on classification accuracy was also investigated. I have shown that when using EEG data, a significant increase in accuracy can be achieved if, in addition to or instead of the raw data, the power measured in the wavebands as an extracted feature is used as input to the MLP learning algorithm. By calculating the absolute power of the EEG signal in the alpha, beta, gamma, delta and theta wavebands and using them as input data, the network achieved a recognition accuracy of 91.5%, while using the raw data directly, the recognition accuracy was 56.2%. An attempt was also made to perform an independent component analysis on the raw data and use the resulting values, and in another approach the absolute power calculated from these values, as input to the neural network. My measurements showed that using ICA alone gives worse results compared to PSD on raw data, and using ICA and PSD together gives similar results compared to PSD alone. Based on these, I do not consider it necessary to perform an independent component analysis.

Using data from the PhysioNet EEG database, I compared the performance of several different MLP and CNN architectures. I found that the automatic feature extraction capability of CNN can be used well in the case of EEG signals, the CNN architectures applied to raw data provided overall better results than the MLP with PSD as input, used in many studies.

Also on the PhysioNet EEG database data, my measurements show that the neural network recognition accuracy can be clearly increased by choosing an optimal architecture. Using only 16 of the available 64 channels of data, the best recognition accuracy measured on test data were 99.1% and 97.7% using

data from 10 and 20 subjects, respectively, for classifications involving three activities. I also found that causeless deepening of a network not only does not further increase but may even decrease the classification accuracy – in addition to the need for more training time.

And finally, as a last step of my research, I investigated the potential of FPGAs for hardware acceleration of a trained neural network. On the one hand, the implementation can be done in a purely hardware way. For this pure hardware implementation, I developed a software capable of generating synthesizable Verilog code for the FPGA implementation of an MLP network. The software performs floating-point to fixed-point conversions based on the weights and offset values exported from the neural network training framework and – after defining the architecture – generates the codes implementing multipliers, adders and activation functions. The generated Verilog codes can be imported and synthesized in any FPGA development environment.

Another possible way to implement hardware acceleration is an approach in which we do not implement the specific network itself in the FPGA, but create a processor specifically optimized for running inference of neural networks (DPU), which is then able to execute the code of a machine learning model compiled into its own instruction set. The latter method is more complex overall, but the DPU itself is available as an IP, and compilers are available for the most popular frameworks, so it can be used well from a user's point of view.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Dr. Oniga Istvánnak a cikkeim elkészítéséhez nyújtott ötletekért, tanácsokért és iránymutatásért, valamint a folyamatos szakmai és mindennapi életben való támogatásáért.

Szeretnék köszönetet mondani Dr. Ujvári Balázsnak és Dr. Dávid Gábornak, akik motiváltak a kutatói pálya elkezdésére, valamint az egyetemi és doktoranduszi éveimet is végig segítették.

Külön köszönöm családomnak és páromnak a türelmet és bátorítást, amit a tanulmányaim során, valamint az értekezésem elkészítéséhez kaptam tőlük.

Emellett köszönöm azt az anyagi támogatást[1,2], amit a kutatásom során kaptam.

[1] A kutatást az „Integrált kutatói utánpótlás-képzési program az informatika és számítástudomány diszciplináris területein” (EFOP-3.6.3-VEKOP-16-2017-00002) című projekt támogatta. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

[2] A SETIT projekt (azonosító: 2018-1.2.1-NKP-2018-00004) a „Nemzeti Kiválósági Program: 2018-1.2.1-NKP” pályázati program keretében, a Nemzeti Kutatási és Innovációs Alapból biztosított támogatással valósul meg.

Irodalomjegyzék

[1] S. Sanei, J.A. Chambers, EEG Signal Processing, John Wiley & Sons, England, 2007. ISBN-13: 9780470025819

[2] R. T. Schirrmeister, J. T. Springenberg, …, T. Ball (2018): Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG. arXiv:1703.05051v5

[3] Nijboer, F., Sellers, E. W., …, Kübler, A. (2008): A P300-based brain-computer interface for people with amyotrophic lateral sclerosis. Clincical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology, 119(8):1909–1916.

[4] Münßinger JI, Halder S, Kleih SC, et al. Brain Painting: First Evaluation of a New Brain-Computer Interface Application with ALS-Patients and Healthy Volunteers. Front Neurosci. 2010;4:182. Published 2010 Nov 22. doi:10.3389/fnins.2010.00182

[5] Tonin, L., Carlson, T., Leeb, R., Millán, J. d. R. (2011). Brain-controlled telepresence robot by motor-disabled people. In 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pages 4227–4230.

[6] C. -S. Wei, Y. -T. Wang, C. -T. Lin and T. -P. Jung, "Toward Drowsiness Detection Using Non-hair-Bearing EEG-Based Brain-Computer Interfaces," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 26, no. 2, pp. 400-406, Feb. 2018, doi: 10.1109/TNSRE.2018.2790359.

[7] P. Handa, M. Mathur, N. Goel (2021): Open and free EEG datasets for epilepsy diagnosis. arXiv:2108.01030

[8] MG Terzano, L Parrino, A Sherieri, R Chervin, S Chokroverty, C Guilleminault, M Hirshkowitz, M Mahowald, H Moldofsky, A Rosa, R Thomas, A Walters. Atlas, rules, and recording techniques for the scoring of cyclic alternating pattern (CAP) in human sleep. Sleep Med 2001 Nov; 2(6):537-553.

[9] Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., ... & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220.

[10] Cho, H., Ahn, M., Ahn, S., Kwon, M., & Jun, S. C. (2017). EEG datasets for motor imagery brain-computer interface. GigaScience, 6(7), 1–8. https://doi.org/10.1093/gigascience/gix034

[11] Bigdely-Shamlo Nima, Mullen Tim, Kothe Christian, Su Kyung-Min, Robbins Kay A. (2015). The PREP pipeline: standardized preprocessing for large-scale EEG analysis. Frontiers in Neuroinformatics, vol. 9. doi: 10.3389/fninf.2015.00016

[12] Mainak Jas, Denis A. Engemann, Yousra Bekhti, Federico Raimondo, Alexandre Gramfort (2017.). Autoreject: Automated artifact rejection for MEG and EEG data. NeuroImage, vol. 159., pp. 417-429. doi: 10.1016/j.neuroimage.2017.06.030.

[13] Marco Congedo, Alexandre Barachant, Rajendra Bhatia. Riemannian geometry for EEG-based braincomputer interfaces; a primer and a review. Brain-Computer Interfaces, Taylor & Francis, 2017, 4 (3), pp.155-174. ff10.1080/2326263X.2017.1297192ff. ffhal-01570120f

[14] E. Parvinnia, M. Sabeti, M. Zolghadri Jahromi, R. Boostani (2014.). Classification of EEG Signals using adaptive weighted distance nearest neighbor algorithm, Journal of King Saud University - Computer and Information Sciences, vol. 26. no 1. pp. 1-6. doi: 10.1016/j.jksuci.2013.01.001.

[15] D. Hari Krishna, I.A. Pasha, T. Satya Savithri (2016). Classification of EEG Motor Imagery Multi Class Signals Based on Cross Correlation. Procedia Computer Science, vol. 85, pp. 490-495. doi: 10.1016/j.procs.2016.05.198.

[16] Chen Z, Wang Y, Song Z. Classification of Motor Imagery Electroencephalography Signals Based on Image Processing Method. Sensors (Basel). 2021;21(14):4646. Published 2021 Jul 7. doi:10.3390/s21144646

[17] Wu, Y.-T., Huang, T. H., Yi Lin, C., Tsai, S. J., & Wang, P.-S. (2018). Classification of EEG Motor Imagery Using Support Vector Machine and Convolutional Neural Network. 2018 International Automatic Control Conference (CACS). doi:10.1109/cacs.2018.8606765

[18] León J, Escobar JJ, Ortiz A, et al. Deep learning for EEG-based Motor Imagery classification: Accuracy-cost trade-off. PLoS One. 2020;15(6):e0234178. Published 2020 Jun 11. doi:10.1371/journal.pone.0234178

[19] Wang Z, Zhang Z, Gong X, Sun Y and Wang H (2018). Short time Fourier transformation and deep neural networks for motor imagery brain computer interface recognition. Concurrency and Computation: Practice and Experience, vol. 30. doi: 10.1002/cpe.4413

[20] Behri, M., Subasi, A., & Qaisar, S. M. (2018). Comparison of machine learning methods for two class motor imagery tasks using EEG in brain-

computer interface. 2018 Advances in Science and Engineering Technology International Conferences (ASET). doi:10.1109/icaset.2018.8376886

[21] Jia, Hongru & Wang, Shuai & Zheng, Dezhi & Qu, Xiaolei & Fan, Shangchun. (2019). Comparative study of motor imagery classification based on BP-NN and SVM. The Journal of Engineering. 2019. 10.1049/joe.2018.9075.

[22] Alexander Craik, Yongtian He, Jose L Contreras-Vidal (2019). Deep learning for electroencephalogram (EEG) classification tasks: a review. Journal of Neural Engineering, vol. 16, no 3. doi: 10.1088/1741-2552/ab0ab5

[23] Amjed S. Al-Fahoum, Ausilah A. Al-Fraihat, "Methods of EEG Signal Features Extraction Using Linear Analysis in Frequency and Time-Frequency Domains", International Scholarly Research Notices, vol. 2014, Article ID 730218, 7 pages, 2014. doi: 10.1155/2014/730218

[24] Swati Aggarwal, Nupur Chugh (2019). Signal processing techniques for motor imagery brain computer interface: A review. Array, vol 1-2, 100003. doi: 10.1016/j.array.2019.100003.

[25] Resalat SN, Saba V. A Study of Various Feature Extraction Methods on a Motor Imagery Based Brain Computer Interface System. Basic Clin Neurosci. 2016;7(1):13-19.

[26] Amjed S. Al-Fahoum, Ausilah A. Al-Fraihat, "Methods of EEG Signal Features Extraction Using Linear Analysis in Frequency and Time-Frequency Domains", International Scholarly Research Notices, vol. 2014, Article ID 730218, 7 pages, 2014 doi: 10.1155/2014/730218

[27] Schalk, G., McFarland, D.J., Hinterberger, T., Birbaumer, N., Wolpaw, J.R. BCI2000: A General-Purpose Brain-Computer Interface (BCI) System. IEEE Transactions on Biomedical Engineering 51(6):1034-1043, 2004.

[28] Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., ... & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220.

[29] Luciw, M., Jarocka, E. & Edin, B. Multi-channel EEG recordings during 3,936 grasp and lift trials with varying weight and friction. Sci Data 1, 140047 (2014). doi: 10.1038/sdata.2014.47

[30] Kaya, Murat; Binli, Mustafa Kemal; Ozbay, Erkan; Yanar, Hilmi; Mishchenko, Yuriy (2018): A large electroencephalographic motor

imagery dataset for electroencephalographic brain computer interfaces. figshare. Collection. https://doi.org/10.6084/m9.figshare.c.3917698.v1

[31] Benjamin Blankertz, Guido Dornhege, Matthias Krauledat, Klaus-Robert Müller, and Gabriel Curio. The non-invasive Berlin Brain-Computer Interface: Fast acquisition of effective performance in untrained subjects. NeuroImage, 37(2):539-550, 2007.

[32] Tangermann, Michael & Müller, Klaus-Robert & Aertsen, Ad & Birbaumer, Niels & Braun, Christoph & Brunner, Clemens & Leeb, Robert & Mehring, Carsten & Miller, Kai & Müller-Putz, Gernot & Nolte, Guido & Pfurtscheller, Gert & Preissl, Hubert & Schalk, Gerwin & Schlögl, Alois & Vidaurre, Carmen & Waldert, Stephan & Blankertz, Benjamin. (2012). Review of the BCI competition IV. Frontiers in neuroscience. 6. 55. 10.3389/fnins.2012.00055.

[33] Schirrmeister, R.T., Springenberg, J.T., Fiederer, L.D.J., Glasstetter, M., Eggensperger, K., Tangermann, M., Hutter, F., Burgard, W. and Ball, T. (2017), Deep learning with convolutional neural networks for EEG decoding and visualization. Hum. Brain Mapp., 38: 5391-5420. https://doi.org/10.1002/hbm.23730

[34] Kemp B, Olivan J. European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data. Clin Neurophysiol. 2003 Sep;114(9):1755-61. doi: 10.1016/s1388-2457(03)00123-8. PMID: 12948806.

[35] Youngjoo Kim, Jiwoo Ryu, Ko Keun Kim, Clive C. Took, Danilo P. Mandic, Cheolsoo Park, "Motor Imagery Classification Using Mu and Beta Rhythms of EEG with Strong Uncorrelating Transform Based Complex Common Spatial Patterns", Computational Intelligence and Neuroscience, vol. 2016, Article ID 1489692, 13 pages, 2016. https://doi.org/10.1155/2016/1489692

[36] Dose, H., Moller, J. S., Puthusserypady, S., & Iversen, H. K. (2018). A Deep Learning MI - EEG Classification Model for BCIs. 2018 26th European Signal Processing Conference (EUSIPCO). doi: 10.23919/eusipco.2018.8553332

[37] Zhang, D., Chen, K., Jian, D., & Yao, L. (2020). Motor Imagery Classification via Temporal Attention Cues of Graph Embedded EEG Signals. IEEE Journal of Biomedical and Health Informatics, 1–1. doi:10.1109/jbhi.2020.2967128

[38] Fadel, W., Kollod, C., Wahdow, M., Ibrahim, Y., & Ulbert, I. (2020). Multi-Class Classification of Motor Imagery EEG Signals Using Image-Based Deep Recurrent Convolutional Neural Network. 2020 8th

International Winter Conference on Brain-Computer Interface (BCI). doi:10.1109/bci48061.2020.9061622

[39] Eitan Netzer, Alex Frid, Dan Feldman (2020). Real-time EEG classification via coresets for BCI applications. Engineering Applications of Artificial Intelligence, vol. 89. Article ID 103455. doi: 10.1016/j.engappai.2019.103455

[40] Tokovarov, M. (2020). Convolutional Neural Networks with Reusable Full-Dimension-Long Layers for Feature Selection and Classification of Motor Imagery in EEG Signals. Lecture Notes in Computer Science, 79–91. doi:10.1007/978-3-030-61609-0_7

[41] Audrey Aldridge, Eli Barnes, Cindy L. Bethel, Daniel W. Carruth, Marianna Kocturova, Matus Pleva, Jozef Juhar, "Accessible Electroencephalograms (EEGs): A Comparative Review with OpenBCI's Ultracortex Mark IV Headset", 29th International Conference Radioelektronika (RADIOELEKTRONIKA), Pardubice, Czech Republic, 2019, pp. 1-6, DOI: 10.1109/RADIOELEK.2019.8733482

[42] Pattisapu S, Ray S. Stimulus-induced narrow-band gamma oscillations in humans can be recorded using open-hardware low-cost EEG amplifier. PLoS One. 2023 Jan 23;18(1):e0279881. doi: 10.1371/journal.pone.0279881. PMID: 36689427; PMCID: PMC9870151.

[43] Gramann, Klaus & Ferris, Daniel & Gwin, Joseph & Makeig, Scott. (2013). Imaging Natural Cognition in Action.. International journal of psychophysiology : official journal of the International Organization of Psychophysiology. 91. 10.1016/j.ijpsycho.2013.09.003.

[44] Gaur, P., Gupta, H., Chowdhury, A., McCreadie, K., Pachori, R. B., & Wang, H. (2021). A Sliding Window Common Spatial Pattern for Enhancing Motor Imagery Classification in EEG-BCI. IEEE Transactions on Instrumentation and Measurement, 70, 1–9. doi:10.1109/tim.2021.3051996

[45] Wang, Xiaying & Hersche, Michael & Tomekce, Batuhan & Kaya, Burak & Magno, Michele & Benini, Luca. (2020). An Accurate EEGNet-based Motor-Imagery Brain–Computer Interface for Low-Power Edge Computing. 1-6. doi: 10.1109/MeMeA49120.2020.9137134.

[46] Blanco-Mora, D. & Aldridge, A. & Jorge, Carolina & Vourvopoulos, Athanasios & Figueiredo, Patrícia & Bermúdez i Badia, Sergi. (2021). Finding the Optimal Time Window for Increased Classification Accuracy during Motor Imagery. 144-151. doi: 10.5220/0010316101440151.

[47] Kalevo, Laura & Myllymaa, Sami & Miettinen, Tomi & Leino, Akseli & Kainulainen, Samu & Korkalainen, Henri & Myllymaa, Katja & Toyras, Juha & Leppanen, Timo & Laitinen, Tiina. (2020). Effect of Sweating on Electrode-Skin Contact Impedances and Artifacts in EEG Recordings With Various Screen-Printed Ag/Agcl Electrodes. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.2977172.

[48] Wada Y, Takizawa Y, Jiang ZY, Yamaguchi N. Gender differences in quantitative EEG at rest and during photic stimulation in normal young adults. Clin Electroencephalogr. 1994 Apr;25(2):81-5. doi: 10.1177/155005949402500209. PMID: 8194192.

[49] He M, Liu F, Nummenmaa A, Hämäläinen M, Dickerson BC, Purdon PL. Age-Related EEG Power Reductions Cannot Be Explained by Changes of the Conductivity Distribution in the Head Due to Brain Atrophy. Front Aging Neurosci. 2021;13:632310. Published 2021 Feb 18. doi:10.3389/fnagi.2021.632310

[50] Özkan, Ali. (2017). Effect of Normalization Techniques on Multilayer Perceptron Neural Network Classification Performance for Rheumatoid Arthritis Disease Diagnosis. International Journal of Trend in Scientific Research and Development. Volume-1. 733-739. 10.31142/ijtsrd3576.

[51] E. Huigen, A. Peper, C. A. Grimbergen, “Investigation into the origin of the noise of surface electrodes”, Medical and Biological Engineering and Computing, vol. 40, pp. 332-338, 2002. DOI: 10.1007/BF02344216

[52] Sampsa Vanhatalo, Juha Voipio, Kai Kaila, “Full-band EEG (FbEEG): an emerging standard in electroencephalography”, Clinical Neurophysiology, vol. 116, pp. 1-8, 2005. DOI: 10.1016/j.clinph.2004.09.015

[53] Alain de Cheveigné, Dorothée Arzounian, “Robust detrending, rereferencing, outlier detection, and inpainting for multichannel data”, NeuroImage, vol. 172, pp. 903-912, 2018. DOI: 10.1016/j.neuroimage.2018.01.035

[54] M. Toscani, T. Marzi, S. Righi, M. P. Viggiano, S. Baldassi: Alpha waves: a neural signature of visual suppression. Exp Brain Res (2010) 207:213–219, DOI 10.1007/s00221-010-2444-7

[55] Renton, A.I., Mattingley, J.B. & Painter, D.R., “Optimising non-invasive brain-computer interface systems for free communication between naïve human participants”, Scientific Reports 9, 18705 (2019). DOI: 10.1038/s41598-019-55166-y

[56] E. Gysels and P. Celka, “Phase synchronization for the recognition of mental tasks in a brain-computer interface”, IEEE Transactions on Neural

Systems and Rehabilitation Engineering, vol. 12, no. 4, pp. 406-415, Dec. 2004, DOI: 10.1109/TNSRE.2004.838443

[57] S. V. Bozhokin and I. B. Suslova, “Wavelet-based analysis of spectral rearrangements of EEG patterns and of non-stationary correlations”, Physica A: Statistical Mechanics and its Applications, vol. 21, pp. 151-160, 2015. DOI: 10.1016/j.physa.2014.11.026

[58] D. Langlois, S. Chartier, and D. Gosselin, “An Introduction to Independent Component Analysis: InfoMax and FastICA algorithms,” Tutorials in Quantitative Methods for Psychology, vol. 6, no. 1, pp. 31–38, 2010.

[59] Scherer, Dominik; Müller, Andreas C.; Behnke, Sven (2010). "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition" (PDF). Artificial Neural Networks (ICANN), 20th International Conference on. Thessaloniki, Greece: Springer. pp. 92–101.

[60] Shenouda, E.A.M.A. (2006). A Quantitative Comparison of Different MLP Activation Functions in Classification. In: Wang, J., Yi, Z., Zurada, J.M., Lu, BL., Yin, H. (eds) Advances in Neural Networks - ISNN 2006. ISNN 2006. Lecture Notes in Computer Science, vol 3971. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11759966_125

[61] Hagan, M.T., and M. Menhaj, “Training feed-forward networks with the Marquardt algorithm,” IEEE Transactions on Neural Networks, Vol. 5, No. 6, 1999, pp. 989–993, 1994.

[62] Pfurtscheller G and Neuper C (1997): Motor imagery activates primary sensorimotor area in humans. Neuros. Letters 239:65-68

[63] N.S. Dias, P.M. Mendes, J.H. Correia (2008): Feature selection for Brain-Computer Interface. IFMBE Proceedings vol. 22/1, pages 318-321

[64] Babiloni C, Carducci F et al. (1999): Human movement-related potentials vs desynchronization of EEG alpha rhytm: a high resolution EEG study. Neuroimage 10:658-6

[65] Saha S, Baumert M. Intra- and Inter-subject Variability in EEG-Based Sensorimotor Brain Computer Interface: A Review. Front Comput Neurosci. 2020 Jan 21;13:87. doi: 10.3389/fncom.2019.00087. PMID: 32038208; PMCID: PMC6985367.

[66] Altuwaijri, Ghadir A., and Ghulam Muhammad. 2022. "A Multibranch of Convolutional Neural Network Models for Electroencephalogram-Based Motor Imagery Classification" Biosensors 12, no. 1: 22. DOI: 10.3390/bios12010022

[67] N. B. Gaikwad, V. Tiwari, A. Keskar and N. C. Shivaprakash, "Efficient FPGA Implementation of Multilayer Perceptron for Real-Time Human Activity Classification," in IEEE Access, vol. 7, pp. 26696-26706, 2019, doi: 10.1109/ACCESS.2019.2900084.

[68] Shakoory, Ghassan. 2013. "FPGA Implementation Of Multilayer Perceptron For Speech Recognition". Journal of Engineering and Sustainable Development, 2013, Volume 17, Issue 6, pp. 175-185

[69] Xilinx, "Alveo U50 Data Center Accelerator Card Data Sheet," DS965 datasheet, Aug. 2020.

[70] Xilinx, "Vitis AI User Guide, " UG1414 (v1.1) datasheet, Mar. 2020.

A. Függelék

1. táblázat. CNN1 konfúziós mátrix (10 fő, 32-es szegmensméret)

Tényleges osztály				
	Kézfej	32501	3299	6120
	Lábfej	2292	34253	4847
	Pihenés	4215	5245	32280
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

2. táblázat. CNN1 konfúziós mátrix (20 fő, 32-es szegmensméret)

Tényleges osztály				
	Kézfej	62831	7637	13061
	Lábfej	10535	59043	14295
	Pihenés	14299	10962	58362
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

3. táblázat. CNN1 konfúziós mátrix (10 fő, 64-es szegmensméret)

Tényleges osztály				
	Kézfej	34661	939	3980
	Lábfej	478	36080	2932
	Pihenés	943	1256	37332
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

4. táblázat. CNN1 konfúziós mátrix (20 fő, 64-es szegmensméret)

Tényleges osztály				
	Kézfej	63268	4044	11887
	Lábfej	3006	66501	9821
	Pihenés	5474	5565	68528
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

5. táblázat. CNN1 konfúziós mátrix (10 fő, 128-as szegmensméret)

Tényleges osztály				
	Kézfej	33211	385	1647
	Lábfej	179	34436	677
	Pihenés	229	230	34704
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

6. táblázat. CNN1 konfúziós mátrix (20 fő, 128-as szegmensméret)

Tényleges osztály				
	Kézfej	69428	505	835
	Lábfej	2577	66034	1934
	Pihenés	4093	1619	65206
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

7. táblázat. CNN1 konfúziós mátrix (10 fő, 160-as szegmensméret)

Tényleges osztály				
	Kézfej	32795	84	176
	Lábfej	76	33014	94
	Pihenés	212	212	32584
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

8. táblázat. CNN1 konfúziós mátrix (20 fő, 160-as szegmensméret)

Tényleges osztály				
	Kézfej	65400	379	786
	Lábfej	560	64773	986
	Pihenés	940	897	64579
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

9. táblázat. CNN2 konfúziós mátrix (20 fő, 32-es szegmensméret)

Tényleges osztály				
	Kézfej	50123	18079	15327
	Lábfej	17264	50931	15678
	Pihenés	20520	17644	45459
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

10. táblázat. CNN2 konfúziós mátrix (20 fő, 64-es szegmensméret)

Tényleges osztály				
	Kézfej	51730	11417	16052
	Lábfej	14095	49350	15883
	Pihenés	15860	12477	51230
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

11. táblázat. CNN2 konfúziós mátrix (20 fő, 128-as szegmensméret)

Tényleges osztály				
	Kézfej	51897	8185	10686
	Lábfej	6368	55017	9160
	Pihenés	9321	10522	51075
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

12. táblázat. CNN2 konfúziós mátrix (20 fő, 160-as szegmensméret)

Tényleges osztály				
	Kézfej	50472	5056	11037
	Lábfej	5649	51491	9179
	Pihenés	8275	7801	50340
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

13. táblázat. CNN3 konfúziós mátrix (20 fő, 32-es szegmensméret)

Tényleges osztály				
	Kézfej	60288	11632	11609
	Lábfej	11352	60933	11588
	Pihenés	17156	15582	50885
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

14. táblázat. CNN3 konfúziós mátrix (20 fő, 64-es szegmensméret)

Tényleges osztály				
	Kézfej	67692	4388	7119
	Lábfej	6955	63303	9070
	Pihenés	11429	8103	60035
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

15. táblázat. CNN3 konfúziós mátrix (20 fő, 128-as szegmensméret)

Tényleges osztály				
	Kézfej	64948	1731	4089
	Lábfej	1557	65070	3918
	Pihenés	4100	3309	63509
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

16. táblázat. CNN3 konfúziós mátrix (20 fő, 160-as szegmensméret)

Tényleges osztály				
	Kézfej	60064	1478	5023
	Lábfej	761	62370	3188
	Pihenés	1046	1215	64155
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

17. táblázat. CNN4 konfúziós mátrix (20 fő, 32-es szegmensméret)

Tényleges osztály				
	Kézfej	59235	12477	11817
	Lábfej	9473	63622	10778
	Pihenés	13889	15851	53883
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

18. táblázat. CNN4 konfúziós mátrix (20 fő, 64-es szegmensméret)

Tényleges osztály				
	Kézfej	63649	6615	8935
	Lábfej	4164	67832	7332
	Pihenés	9715	11269	58583
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

19. táblázat. CNN4 konfúziós mátrix (20 fő, 128-as szegmensméret)

Tényleges osztály				
	Kézfej	67792	1094	1882
	Lábfej	2062	66125	2358
	Pihenés	4842	2808	63268
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

20. táblázat. CNN4 konfúziós mátrix (20 fő, 160-as szegmensméret)

Tényleges osztály				
	Kézfej	64481	766	1318
	Lábfej	711	64549	1059
	Pihenés	2283	1586	62547
		Kézfej	Lábfej	Pihenés

Modell általi besorolás

B. Függelék: Publikációs lista

Idegen nyelvű folyóiratcikkek

[J1] Majoros, T.; Oniga, S. Overview of the EEG-Based Classification of Motor Imagery Activities Using Machine Learning Methods and Inference Acceleration with FPGA-Based Cards. Electronics 2022, 11, 2293. DOI: 10.3390/electronics11152293

[J2] Tamás Majoros, Stefan Oniga, Yu Xie, Motor imagery EEG classification using feedforward neural network, Annales Mathematicae et Informaticae vol 53 (2021), pp. 235-244, ISSN 1787 - 6117, DOI: 10.33039/ami.2021.04.007

[J3] Tamás Majoros, Balázs Ujvári, Stefan Oniga, EEG data processing with neural network, Carpathian Journal of Electronic and Computer Engineering vol 12 number 2/2019, pp. 33-36, ISSN 1844 – 9689, DOI: 10.2478/cjece-2019-0014

[J4] Tamás Majoros, Balázs Ujvári, Stability study of the neural network at particle physics detectors, Carpathian Journal of Electronic and Computer Engineering vol 11 number 1/2018, pp. 48-52, ISSN 1844 – 9689, DOI: 10.2478/cjece-2018-0009

[J5] C. A. Aidala, …, T. Majoros, …, S. Polizzo, Design and Beam Test Results for the sPHENIX Electromagnetic and Hadronic Calorimeter Prototypes, IEEE Transactions on Nuclear Science vol 65 issue 12, pp. 2901-2919, ISSN 0018-9499, DOI: 10.1109/TNS.2018.2879047

[J6] B. Biró, …, T. Majoros, …, C. L. Woody, A Comparison of the Effects of Neutron and Gamma Radiation in Silicon Photomultipliers, IEEE Transactions on Nuclear Science vol 66 issue 7, pp. 1833-1839, ISSN 0018-9499, DOI: 10.1109/TNS.2019.2921102

[J7] Xie, Y.; Majoros, T.; Oniga, S. FPGA-Based Hardware Accelerator on Portable Equipment for EEG Signal Patterns Recognition. Electronics 2022, 11, 2410. DOI: 10.3390/electronics11152410

Idegen nyelvű konferenciacikkek

[C1] T. Majoros and S. Oniga, "Comparison of Motor Imagery EEG Classification using Feedforward and Convolutional Neural Network," IEEE EUROCON 2021 - 19th International Conference on Smart Technologies, 2021, pp. 25-29, doi: 10.1109/EUROCON52738.2021.9535592.

[C2] T. Majoros and S. Oniga, "Activity recognition using consumer-grade EEG device," 2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), 2021, pp. 1-6, doi: 10.1109/ECAI52376.2021.9515106.

[C3] X. Yu, T. Majoros and S. Oniga, "Hardware Implementation of CNN Based on FPGA for EEG Signal Patterns Recognition," 2021 International Conference on e-Health and Bioengineering (EHB), 2021, pp. 1-4, doi: 10.1109/EHB52898.2021.9657679.

[C4] Yu Xie, Stefan Oniga, Tamás Majoros, Comparison of EEG Data Processing Using Feedforward and Convolutional Neural Network, 1st Conference on Information Technology and Data Science, Debrecen, Hungary, November 6–8, 2020, pp. 279-289, ISSN 1613-0073

[C5] L. Kovács, ...,T. Majoros, T. Bérczes, "Sensor design and integration into small sized autonomous vehicle," 2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS), 2022, pp. 171-176, doi: 10.1109/CITDS54976.2022.9914037.

Magyar nyelvű konferenciacikkek

[C6] Majoros Tamás, Oniga István László, Elképzelt motoros tevékenységek EEG-alapú osztályozása neurális hálózatok használatával, XXIII. NEMZETKÖZI SZÁMÍTÁSTECHNIKA ÉS OKTATÁS KONFERENCIA, SzámOkt 2022, Marosvásárhely, október 13-16, 2022, pp. 162-167, ISSN 2734-6757

[C7] Majoros Tamás, Ujvári Balázs, Oniga István László, EEG adatok feldolgozása neurális hálózattal, XXIX. NEMZETKÖZI SZÁMÍTÁSTECHNIKA ÉS OKTATÁS KONFERENCIA, SzámOkt 2019, Temesvár, október 10-13, 2019, pp. 208-213, ISSN 1842 - 4546

[C8] Majoros Tamás, Ujvári Balázs, Oniga István László, Részecskefizikai detektorban alkalmazott neurális hálózat stabilitási vizsgálata, XXVIII.

NEMZETKÖZI SZÁMÍTÁSTECHNIKA ÉS OKTATÁS KONFERENCIA, SzámOkt 2018, Tusnádfürdő, október 11-14, 2018, pp. 262-267, ISSN 1842 - 4546